PCT/JP00/05135

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

31.07.00

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

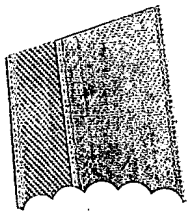
出 願 年 月 日
Date of Application: １９９９年 ７月２９日

出 願 番 号
Application Number: 平成１１年特許願第２１５４５０号

出 願 人
Applicant (s): ターボデータラボラトリー有限会社

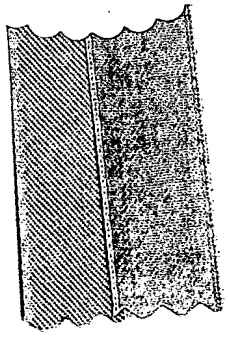
JP00/05135

王木山



PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)



２０００年 ９月 １日

特許庁長官
Commissioner,
Patent Office 及川耕造

出証番号 出証特２０００－３０６９０３５

【書類名】　特許願

【整理番号】　PK990013

【あて先】　特許庁長官　殿

【国際特許分類】　G06F 17/30

【発明者】

【住所又は居所】　神奈川県横浜市神奈川区松見町４丁目１１０１番地７コートハウス菊名８０４号

【氏名】　古庄　晋二

【特許出願人】

【住所又は居所】　東京都台東区松が谷１－９－１２　ＳＰＫビルディング６０４号

【氏名又は名称】　ターボデータラボラトリー有限会社

【代表者】　古庄　晋二

【代理人】

【識別番号】　100103632

【弁理士】

【氏名又は名称】　窪田　英一郎

【選任した代理人】

【識別番号】　100099715

【弁理士】

【氏名又は名称】　吉田　聡

【手数料の表示】

【予納台帳番号】　058377

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　表形式データの提示方法、挿入方法、削除方法、更新方法およびこれら方法を利用したトランザクション処理方法、並列処理方法、並びに、上記方法を実現するプログラムを記憶した記憶媒体

【特許請求の範囲】

【請求項１】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの任意の位置に、項目値を挿入する表形式データの挿入方法であって、

レコード番号を添え字として受け入れ、当該添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、

挿入される項目値の位置を示す挿入位置を特定し、

前記添え字変換配列において、前記挿入位置に関して、対応する添え字の範囲を確定するとともに、前記配列の末尾以降の所定の位置を特定するためのオフセット値を与え、

前記添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を上方にシフトするとともに、受け入れられた添え字をデクリメントするためのオフセット値を与え、

前記配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データの挿入方法。

【請求項２】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意の位置の項目値を削除する表形式データの削除方法であって、

レコード番号を添え字として受け入れ、受け入れられた添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、

削除すべき項目値の位置を示す削除位置を特定し、

前記添え字変換配列において、削除位置に対応するレコード番号より大きなレ

コード番号を有するものに関して、対応する添え字の範囲を下方にシフトするとともに、受け入れられた添え字をインクリメントするためのオフセット値を与え、

前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データの削除方法。

【請求項３】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意の位置の項目値を更新する表形式データの更新方法であって、

レコード番号を添え字として受け入れ、受け入れられた添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、

（１）更新すべき項目値の位置を削除位置として仮定して、

前記添え字変換配列において、削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を下方にシフトするとともに、受け入れられた添え字をインクリメントするためのオフセット値を与え、

（２）更新すべき項目値の位置を挿入位置として仮定して、

前記添え字変換配列において、前記挿入位置に関して、対応する添え字の範囲を確定するとともに、前記配列の末尾以降の所定の位置を特定するためのオフセット値を与え、

前記添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を上方にシフトするとともに、受け入れられた添え字をデクリメントするためのオフセット値を与え、

かつ、前記配列の末尾以降の所定の位置に、前記更新すべき項目値を配置し、

前記（１）および（２）、或いは、（２）および（１）を順次実行して、

前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データの更新方法。

【請求項４】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの項目値の値を変換する表形式データの値変換方法であって、

前記項目値の範囲に対応するオフセット値を与えるように構成された値変換配列を生成し、

レコード番号を添え字として受け入れ、前記配列中の、当該添え字に対応する項目値に、前記項目値の範囲に対応したオフセット値を与えるように構成されたことを特徴とする表形式データの値変換方法。

【請求項５】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意のレコード番号に対応する項目値を特定するように構成された表形式データのデータ構造であって、

当該表形式データを、特定の項目に属する項目値に対応した項目値番号の順に当該項目値が格納されている第１の実体配列を含む値リストと、一意的なレコード番号の順に、当該項目値番号を指示するためのポインタ値が格納された第２の実体配列を含むポインタ配列とからなる一以上の情報ブロックに分割するように構成し、

前記ポインタ配列に、入力されたレコード番号を添え字として、当該添え字の範囲に対応する第１のオフセットを与えるように構成された第１の添え字変換配列を形成するとともに、当該添え字変換配列を経てオフセットが与えられた添え字により特定される、第２の実体配列中のポインタ値を値として、当該値の範囲に対応する第２のオフセットを与えるように構成された値変換配列を形成し、

前記値リストに、入力された、ポインタ配列の出力を添え字として、当該添え字の範囲に対応する第３のオフセットを与えるように構成された第２の添え字変換配列を形成し、当該第２の添え字変換配列を経てオフセットが与えられたポインタ配列の出力により、第１の実体配列中の項目値が特定されるように構成されたことを特徴とする表形式データのデータ構造。

【請求項６】　前記添え字変換配列の各々が、所定の範囲にそれぞれ含まれる添え字の最小値を示す開始位置からなる開始位置配列、および／または、当該所定の範囲に含まれる添え字の最大値を示す終了位置からなる終了位置配列と、対

応するオフセット値からなるオフセット配列とから構成されることを特徴とする請求項５に記載のデータ構造。

【請求項７】　前記値変換配列が、所定の範囲にそれぞれ含まれる値の最小値を示す開始位置からなる開始位置配列、および／または、当該所定の範囲に含まれる値の最大値を示す終了位置からなる終了位置配列と、対応するオフセット値からなるオフセット配列とから構成されることを特徴とする請求項５に記載のデータ構造。

【請求項８】　請求項５ないし７の何れか一項に記載のデータ構造にて表わされる表形式データの任意の位置に、項目値を挿入する表形式データの挿入方法であって、

（１）前記値リストに関して、

挿入される項目値の位置を示す挿入位置を特定し、

前記第２の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第３のオフセット値を与え、

前記第２の添え字変換配列において、前記挿入位置に対応する添え字の値より大きなものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第３のオフセット値を与え、

前記第１の実体配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

（２）前記ポインタ配列に関して、

挿入されるレコード番号に対応するポインタ値の位置を示す挿入位置を特定し、

前記第１の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第１のオフセット値を与え、

前記第１の添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範

囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第１のオフセット値を与え、

前記第２の実体配列の末尾以降の所定の位置に、現存するポインタ値より大きな新たなポインタ値を配置し、かつ、

（３）前記ポインタ配列に関して、

前記値変換配列に、前記値リストにおける挿入位置に対応するポインタ値以上を有するものに対して、これをインクリメントする第２のオフセット値を与え、

前記値変換配列に、前記新たなポインタ値が、前記挿入位置に対応する位置を特定するような第２のオフセット値を与えることを特徴とする表形式データのデータ挿入方法。

【請求項９】　ｍ個のレコード番号と、ｎ個の項目値とを備え、

前記値リスト中の項目値の挿入位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾ｎに項目値が配置され、かつ、レコード番号の挿入位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾ｍにポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、

値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、

値がｉである場合に、第３のオフセット値として（ｎ－ｉ）を与え、かつ、

値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、

添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、

添え字がｊである場合に、第１のオフセット値として（ｍ－ｊ）を与え、かつ、

添え字が（ｊ＋１）以上ｍ以下の場合に、第１のオフセット値として（－１）を与え、

（３）前記ポインタ配列の値変換配列において、

値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、

値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、

値がｎである場合に、第２のオフセット値として（ｉ－ｎ）を与えることを特徴とする請求項８に記載の表形式データの挿入方法。

【請求項１０】　ｍ個のレコード番号と、ｎ個の項目値とを備え、

前記値リスト中の項目値の挿入位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾以降の所定の位置ｚ（ｚ≧ｎ）に項目値が配置され、かつ、レコード番号の挿入位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾以降の所定の位置ｘ（ｘ≧ｍ）にポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、

値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、

値がｉである場合に、第３のオフセット値として（ｚ－ｉ）を与え、かつ、

値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、

添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、

添え字がｊである場合に、第１のオフセット値として（ｘ－ｊ）を与え、かつ、

添え字が（ｊ＋１）以上ｍ以下の場合に、第１のオフセット値として（－１）を与えるとともに、

（３）前記ポインタ配列の値変換配列において、

値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、

値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、

値がｙ（ただし、ｙは、第２の実体配列の位置ｘに格納されたポインタ値）である場合に、第２のオフセット値として（ｉ－ｙ）を与えることを特徴とする請求項８に記載の表形式データの挿入方法。

【請求項１１】　請求項５ないし７の何れか一項に記載のデータ構造にて表わ

される表形式データの任意の位置の項目値を削除する表形式データの削除方法であって、

前記ポインタ配列に関して、

削除されるポインタ値の位置を示す削除位置を特定し、

前記第１の添え字変換配列において、前記削除に関して、対応する添え字を確定し、

前記第１の添え字変換配列において、前記削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第１のオフセット値を与えることを特徴とする表形式データの削除方法。

【請求項１２】　ｍ個のレコード番号と、ｎ個の項目値とを備え、

レコード番号の削除位置がｊ（０≦ｊ≦ｍ－１）である場合に、

前記ポインタ配列の第１の添え字変換配列において、

添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、かつ、

添え字の範囲がｊ以上（ｍ－２）以下である場合に、オフセット値として１を与えることを特徴とする請求項１１に記載の表形式データの削除方法。

【請求項１３】　請求項５ないし７の何れか一項に記載のデータ構造にて表わされる表形式データの任意の位置の項目値を更新する表形式データの更新方法であって、

（Ａ）（１）前記値リストに関して、

更新される項目値を挿入される項目値と仮定し、当該項目値の挿入すべき位置を示す挿入位置を特定し、

前記第２の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するためのオフセット値を与え、

前記第２の添え字変換配列において、前記挿入位置に対応する添え字の値より大きなものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きく

なるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第３のオフセット値を与え、

前記第１の実体配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

（２）前記ポインタ配列に関して、

更新されるポインタ値を挿入されるポインタ値と仮定し、当該ポインタ値の挿入すべき位置を示す挿入位置を特定し、

前記第１の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第１のオフセット値を与え、

前記第１の添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第１のオフセット値を与え、

前記第２の実体配列の末尾以降の所定の位置に、現存するポインタ値より大きな新たなポインタ値を配置し、かつ、

（３）前記ポインタ配列に関して、

前記値変換配列に、前記値リストにおける挿入位置に対応するポインタ値以上を有するものに対して、これをインクリメントする第２のオフセット値を与え、

前記値変換配列に、前記新たなポインタ値が、前記挿入位置に対応する位置を特定するような第２のオフセット値を与え、

(Ｂ)前記ポインタ配列に関して、

更新すべきポインタ値の位置を、削除位置として、前記挿入位置を考慮して特定し、

前記第１の添え字変換配列において、前記削除位置に関して、対応する添え字を確定し、

前記第１の添え字変換配列において、前記削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた

添え字をインクリメントするための第１のオフセット値を与え、
前記(A)および(B)、或いは、前記(B)および(A)の何れかを順次実行するように構成されたことを特徴とする表形式データの更新方法。

【請求項１４】　ｍ個のレコード番号と、ｎ個の項目値とを備え、

前記値リスト中の項目値の更新位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾ｎに項目値が配置され、かつ、レコード番号の更新位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾ｍにポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、

値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、

値がｉである場合に、第３のオフセット値として（ｎ－ｉ）を与え、かつ、

値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、

添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、

添え字がｊである場合に、第１のオフセット値として（ｍ－ｊ）を与え、かつ、

添え字が（ｊ＋１）以上（ｍ＋１）以下の場合に、第１のオフセット値として０を与え、

（３）前記ポインタ配列の値変換配列において、

値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、

値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、

値がｎである場合に、第２のオフセット値として（ｉ－ｎ）を与えることを特徴とする請求項１３に記載の表形式データの更新方法。

【請求項１５】　ｍ個のレコード番号と、ｎ個の項目値とを備え、

前記値リスト中の項目値の更新位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾以降の所定の位置ｚ（ｚ≧ｎ）に更新された項目値が配置され、

かつ、レコード番号の更新位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾以降の所定の位置ｘ（ｘ≧ｍ）に更新されたポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、

値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、

値がｉである場合に、第３のオフセット値として（ｚ－ｉ）を与え、かつ、

値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、

添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、

添え字がｊである場合に、第１のオフセット値として（ｘ－ｊ）を与え、かつ、

添え字が（ｊ＋１）以上（ｍ＋１）以下の場合に、第１のオフセット値として０を与えるとともに、

（３）前記ポインタ配列の値変換配列において、

値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、

値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、

値がｙ（ただし、ｙは、第２の実体配列の位置ｘに格納されたポインタ値）である場合に、第２のオフセット値として（ｉ－ｙ）を与えることを特徴とする請求項１３に記載の表形式データの挿入方法。

【請求項１６】　請求項５ないし７の何れか一項に記載のデータ構造にて表わされる表形式データの任意の位置の項目値を削除する表形式データの削除方法であって、

前記レコード番号に関して、レコード番号自体を配置した第４の実体配列と、当該レコード番号を特定するための添え字の範囲にしたがって、所定のオフセット値を与える第３の添え字変換配列とを設け、

前記削除すべきレコード番号の位置を示す削除位置を特定し、

前記第３の添え字変換配列において、前記削除に関して対応する添え字を確定し、

前記第３の添え字変換配列において、前記削除位置に対応する添え字より大きな値を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第４のオフセット値を与えることを特徴とする表形式データの削除方法。

【請求項１７】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの任意の位置の項目値を削除する表形式データの削除方法であって、

前記レコード番号に関して、レコード番号自体を配置した配列と、当該レコード番号を特定するための添え字の範囲にしたがって、所定のオフセット値を与える添え字変換配列とを設け、

前記削除すべきレコード番号の位置を示す削除位置を特定し、

前記添え字変換配列において、前記削除に関して対応する添え字を確定し、

前記添え字変換配列において、前記削除位置に対応する添え字より大きな値を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするためのオフセット値を与えることを特徴とする表形式データの削除方法。

【請求項１８】　請求項８ないし１５の何れか一項に記載の方法により表形式データにおけるデータの挿入、削除および／または更新をなし、ロールバックおよびコミットの何れかの場合には、前記添え字変換配列および値変換配列を廃棄することを特徴とするトランザクション処理方法。

【請求項１９】　請求項８ないし１５の何れか一項に記載の方法により表形式データにおけるデータの挿入、削除および／または更新をなし、ロールバックの場合に、前記添え字変換配列および値変換配列を廃棄することを特徴とするトランザクション方法。

【請求項２０】　請求項５ないし７のデータ構造を有する表形式データに関して、複数のプロセスにより同時に処理が実行されるプロセスの並列処理方法であ

って、

前記プロセスのうち、データの挿入、削除および／または更新を伴うものは、前記ポインタ配列において、第１の添え字変換配列、第２の実体配列および値変換配列を経るように構成されるとともに、前記値リストにおいて、第２の添え字変換配列および第１の実体配列を経るように構成され、

その一方、前記プロセスのうち、データの挿入、削除および／または更新を伴わないものは、前記ポインタ配列において、第２の実体配列を経るとともに、前記値リストにおいて、前記値リストにおいて、第１の実体配列を経るように構成されたことを特徴とするプロセスの並列処理方法。

【請求項２１】　前記データの挿入、削除および／または更新を伴うプロセスが、請求項８ないし請求項１５の何れか一項に記載された方法を利用することを特徴とする請求項２０に記載のプロセス並列処理方法。

【請求項２２】　請求項５ないし７のデータ構造を有する表形式データに関して、レコードのロックをなす方法であって、

ロックを管理する配列を含む情報ブロックを設け、

前記情報ブロックの配列中に、前記レコード番号の各々に対応して、ロックの種別を示す項目値を配置することを特徴とする方法。

【請求項２３】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの任意の位置に、項目値を挿入する表形式データのデータ挿入プログラムを記憶した、コンピュータにより読み出し可能な記憶媒体であって、

レコード番号を添え字として受け入れ、当該添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、

挿入される項目値の位置を示す挿入位置を特定し、

前記添え字変換配列において、前記挿入位置に関して、対応する添え字の範囲を確定するとともに、前記配列の末尾を特定するためのオフセット値を与え、

前記添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を上方にシフトするとともに、受け入れられた添え字をデクリメントするためのオフセット値を与

え、

前記配列の末尾に、前記挿入すべき項目値を配置し、

前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データのデータ挿入プログラムを記憶した記憶媒体。

【請求項２４】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意の位置の項目値を削除する表形式データのデータ削除プログラムを記憶した、コンピュータにより読み出し可能な記憶媒体であって、

レコード番号を添え字として受け入れ、受け入れられた添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、

削除すべき項目値の位置を示す削除位置を特定し、

前記添え字変換配列において、削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を下方にシフトするとともに、受け入れられた添え字をインクリメントするためのオフセット値を与え、

前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データのデータ削除プログラムを記憶した記憶媒体。

【請求項２５】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意の位置の項目値を更新する表形式データのデータ更新プログラムを記憶した、コンピュータにより読み出し可能な記憶媒体であって、

レコード番号を添え字として受け入れ、受け入れられた添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、

（１）更新すべき項目値の位置を削除位置として仮定して、

前記添え字変換配列において、削除位置に対応するレコード番号より大きなレ

コード番号を有するものに関して、対応する添え字の範囲を下方にシフトするとともに、受け入れられた添え字をインクリメントするためのオフセット値を与え、

（２）更新すべき項目値の位置を挿入位置として仮定して、

前記添え字変換配列において、前記挿入位置に関して、対応する添え字の範囲を確定するとともに、前記配列の末尾を特定するためのオフセット値を与え、

前記添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を上方にシフトするとともに、受け入れられた添え字をデクリメントするためのオフセット値を与え、

かつ、前記配列の末尾に、前記更新すべき項目値を配置し、

前記（１）および（２）、或いは、（２）および（１）を順次実行して、

前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データのデータ更新プログラムを記憶した記憶媒体。

【請求項２６】　項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの項目値の値を変換する表形式データの値変換プログラムを記憶した、コンピュータにより読み出し可能な記憶媒体であって、

項目値の範囲に対応するオフセット値を与えるように構成された値変換配列を生成し、

レコード番号を添え字として受け入れ、前記配列中の、当該添え字に対応する項目値に、前記項目値の範囲に対応したオフセット値を与えるように構成されたことを特徴とする表形式データの値変換プログラムを記憶した記憶媒体。

【請求項２７】　請求項５ないし７の何れか一項に記載のデータ構造にて表わされる表形式データの任意の位置に、項目値を挿入する表形式データのデータ挿入プログラムを記憶したコンピュータにより読み出し可能な記憶媒体であって、

（１）前記値リストに関して、

挿入される項目値の位置を示す挿入位置を特定し、

前記第２の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第３のオフセット値を与え、

前記第２の添え字変換配列において、前記挿入位置に対応する添え字の値より大きなものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第３のオフセット値を与え、

前記第１の実体配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

（２）前記ポインタ配列に関して、

挿入されるレコード番号に対応するポインタ値の位置を示す挿入位置を特定し、

前記第１の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第１のオフセット値を与え、

前記第１の添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第１のオフセット値を与え、

前記第２の実体配列の末尾以降の所定の位置に、現存するポインタ値より大きな新たなポインタ値を配置し、かつ、

（３）前記ポインタ配列に関して、

前記値変換配列に、前記値リストにおける挿入位置に対応するポインタ値以上を有するものに対して、これをインクリメントする第２のオフセット値を与え、

前記値変換配列に、前記新たなポインタ値が、前記挿入位置に対応する位置を特定するような第２のオフセット値を与えることを特徴とする表形式データのデータ挿入プログラムを記憶した記憶媒体。

【請求項２８】　請求項５ないし７の何れか一項に記載のデータ構造にて表わされる表形式データの任意の位置の項目値を削除する表形式データのデータ削除

プログラムを記憶した、コンピュータにより読み出し可能な記憶媒体であって、

前記ポインタ配列に関して、

削除されるポインタ値の位置を示す削除位置を特定し、

前記第１の添え字変換配列において、前記削除に関して、対応する添え字を確定し、

前記第１の添え字変換配列において、前記削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第１のオフセット値を与えることを特徴とする表形式データのデータ削除プログラムを記憶した記憶媒体。

【請求項２９】　請求項５ないし７の何れか一項に記載のデータ構造にて表わされる表形式データの任意の位置の項目値を更新する表形式データのデータ更新プログラムを記憶した、コンピュータにより読み出し可能な記憶媒体であって、

（Ａ）（１）前記値リストに関して、

更新される項目値を挿入される項目値と仮定し、当該項目値の挿入すべき位置を示す挿入位置を特定し、

前記第２の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するためのオフセット値を与え、

前記第２の添え字変換配列において、前記挿入位置に対応する添え字の値より大きなものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第３のオフセット値を与え、

前記第１の実体配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

（２）前記ポインタ配列に関して、

更新されるポインタ値を挿入されるポインタ値と仮定し、当該ポインタ値の挿入すべき位置を示す挿入位置を特定し、

前記第１の添え字変換配列において、前記挿入位置に関して、対応する添え字

を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第１のオフセット値を与え、

前記第１の添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第１のオフセット値を与え、

前記第２の実体配列の末尾以降の所定の位置に、現存するポインタ値より大きな新たなポインタ値を配置し、かつ、

（３）前記ポインタ配列に関して、

前記値変換配列に、前記値リストにおける挿入位置に対応するポインタ値以上を有するものに対して、これをインクリメントする第２のオフセット値を与え、

前記値変換配列に、前記新たなポインタ値が、前記挿入位置に対応する位置を特定するような第２のオフセット値を与え、

(Ｂ)前記ポインタ配列に関して、

更新すべきポインタ値の位置を、削除位置として、前記挿入位置を考慮して特定し、

前記第１の添え字変換配列において、前記削除位置に関して、対応する添え字を確定し、

前記第１の添え字変換配列において、前記削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第１のオフセット値を与え、

前記(Ａ)および(Ｂ)、或いは、前記(Ｂ)および(Ａ)の何れかを順次実行するように構成されたことを特徴とする表形式データのデータ更新プログラムを記憶した記憶媒体。

【発明の詳細な説明】

【０００１】

【産業上の技術分野】

本発明は、コンピュータのような情報処理装置を用いて大量のデータを処理す

るデータ処理方法及びデータ処理装置に関し、より詳細には、データベースを構成する表形式データの更新、削除、挿入およびトランザクション処理に関する。

【０００２】

【従来の技術】

データベースは種々の用途に用いられているが、中規模ないし大規模システムにおいては、論理的な矛盾が排除できるリレーショナルデータベース（ＲＤＢ）の使用が主流となっている。たとえば、ＲＤＢは飛行機の座席予約等のシステムに利用されている。この場合、キー項目を指定することにより、（多くの場合１件の）ターゲットを迅速に検索することもでき、或いは、予約の確定、キャンセル或いは変更などを行うことができる。

しかし、近年、データベースを、上記の如く単純に個々のレコードデータを管理する用途（基幹系業務）以外に、分析（情報系業務） にも使用する傾向が顕著である。例えば、上記データベースの所有者は、時間帯、路線、季節などによって搭乗率がどう変化するかを、クロス集計を作成して分析することを試みるであろう。

【０００３】

【発明が解決しようとする課題】

ところが、従来のＲＤＢは、ローカルメモリを使用しコミットすることによりその内容を確定しているため、トランザクション処理を主体とする基幹業務には適しても、その構造上、大規模データの分析（情報系）処理には向かない。たとえば、任意の項目でソートを行おうとすれば、理論上、（たとえばｎ件のレコードでは）、ｎ＊ｌｏｇ（ｎ）の時間がかかり、大規模なデータほど処理効率が低下する。（現実の製品では、理論よりもさらに急速に処理性能の低下が進む。）そして、この制約（＝ｎ＊ｌｏｇ（ｎ）の時間）はソートに限らず、部分集合の抽出（＝検索）など一般的な操作全般に出現する。

【０００４】

もちろん、この制約を回避すべく、さまざまな高速化方法が考案されてはいる。その好例としてビットマップインデックスがある。ビットマップインデックスは、各レコードが条件を満たすか否かを１ビットであらわしたビットマップであ

り、例えば「性別」という項目があった場合、「男性」のビットマップと、「女性」のビットマップが作成される。このとき、「女性」のレコードのみを取り出そうとすれば（つまり、検索すれば）、高速にその部分集合（すなわち、女性のレコードの集合）を得ることができる。しかし、各値毎にビットマップを作成するということはいつでも可能なわけではない。例えば、１０億レコードで１０億通りの値をとる項目（ＩＤ番号など）の場合、１０億ビット（約１２５ＭＢ）のビットマップを１０億個作成しなければならなくなる。つまり、ビットマップインデックスは非常に少数のバリエーションしか持たない項目（たとえば性別）でしか実現できないことが分かる。また、この方法は部分集合に対しては即座には適用できないことが明らかであるし、レコードの挿入・削除が頻発する場合にも使用できない。他の高速化手法も同様に何らかの制約を持っている。このような各種高速化技術の適用範囲の制約のため、ユーザーには全く同等に見える操作が、瞬時に終了する場合と何時間もかかる場合があり、処理時間の極端な非連続性（＝予測困難さ）という深刻な問題が発生している。

たとえば、現在商業的に入手可能なＲＤＢで全体集合が１０００万レコードを有するテーブルで測定したところ、全件（＝１０００万件）をヒットさせた場合、１秒しかかからなかったのに、わずか７３７件をヒットさせただけで６１秒を要したという報告がある。では、９００万件ヒットする場合、どのくらいの処理時間がかかるのであろう。言うまでもないことだが、検索を実行する前に予めヒットする件数を知ることはできないから、ヒット件数と検索の所要時間の対応関数が分かっていたとしても、所要時間を確実に予測することはできない。そもそも、男性という条件を満たす５００万件のレコードを抽出する時間（検索時間）と、女性という条件を満たす５００万件のレコードを抽出する時間（検索時間）が同じであるという理論的根拠すらも希薄ではある。

【０００５】

このようなパフォーマンスの不安定さは、特にリアルタイム性を要求される制御システムにおいては致命的であり、例えば複数の発電所と非常に多数の電力需要を電力線ネットワークで接続した大規模かつ複雑な電力供給制御システム、また例えば大規模な化学プラントの制御システムなど、多量のデータを管理・使用

するシステムでありながらデータベースを使用できない例は数多いのである。

もちろん、パフォーマンスが安定していても遅すぎれば使い物にならないことは明らかであるが、ここで強調したいのは、特殊な限定条件下のみでしか適用できないさまざまな高速化方法が導入された結果、データベースシステムはひどく複雑になったばかりか、前述の処理時間の極端な非連続性という重大な問題を抱え込んだということである。

【０００６】

さて、近年、前述のデータベースの分析的使用方法が一般化していることは前述のようであり、同時にデータの巨大化が急速に進行していることも周知の事実である。このような状況のなかで、ＲＤＢにおける前述の問題を克服するために、ＤＷＨ（データウェアハウス）および多次元ＤＢ（ＭＤＢ：Multi Dimensional Database）が出現した。

ＤＷＨは、トランザクション処理機能などを外し、検索・集計性能に特化したチューニングを行ったＲＤＢであると考えて差し支えない。したがって、ｎ＊ｌｏｇ（ｎ）の処理時間という根本的制約は残存したままであり、パフォーマンスは改善されたものの十分とは言えず、その非連続性の問題も解消されていない。さらに、基幹系の既存ＲＤＢと情報系のＤＷＨという不可避な組み合わせにより、設備投資が大幅に増加し、情報管理の複雑化と運用コスト増加という新たな問題が発生している。

【０００７】

一方、ＭＤＢは指定された項目（次元とよばれている）の組み合わせによる基礎クロス集計を予め作成しておき、ユーザーの要求するクロス集計を基礎クロス集計から合成するものであると考えて良い。ＭＤＢは、クロス集計こそは満足なレスポンスで合成できるが、問題も多い。例えば、（集計から元データに戻ることは不可能なので、特別な工夫をしない限り）元々のレコードに戻ることができない、（予めクロス集計を作成しておかねばならないため、）株価などの時事刻々と更新されるデータに対応できない、あるいは、予め指定した以外の項目（次元）のクロス集計は作成できないことなどの問題が挙げられる。もちろん、設備投資の増加、情報管理の複雑化と運用コストの上昇というＤＷＨと同じ問題も発

生している。

現実を見ると、ＤＷＨやＭＤＢは、前述のような制御システム、あるいは科学技術計算、金融工学など、コストを度外視しても性能を重視する分野においてすら稀にしか導入されない。このことは、これらの機能及び性能が前述の分野において実用に耐えないことの証左である。

【０００８】

このような問題を解決すべく、本発明者はすでに従来技術よりも略１００～１０００倍高速に表形式データを検索・集計・ソート・ジョインする方法（線形フィルター法）を考案し、これに関する特許出願をなしている（（特願平１０－２７２７８号）。線形フィルター法は、処理性能の高さと安定性、処理の統一性、データのコンパクトさと分割性、Ｏ（ｎ）のアーキテクチャーであること、ソートの直列接続性、パイプライン処理や並行処理に適することなど非常に好ましい性質を多々備えている。

そこで、本発明は、特に、上記線形フィルター法を用いて、データの挿入、削除および更新の処理を高速かつ適切になすことができる表形式データにおけるデータ挿入、削除および更新方法を提供することを目的とする。

【０００９】

また、広く一般に理解されているとおり、更新処理の実運用においては、複数のプロセスによる同一データへの同時アクセスを制御する排他処理、複数の処理を１つの有意義な処理ブロックとして一括してコミット（またはロールバック）するトランザクション処理を適切になすことが要求される。

そこで、本発明は、上記トランザクション処理を適切になすことができる表形式データの挿入、削除および更新方法を提供することを目的とする。

【００１０】

【課題を解決するための手段】

本発明の目的は、項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの任意の位置に、項目値を挿入する表形式データの挿入方法であって、レコード番号を添え字として受け入れ、当該添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、挿入され

る項目値の位置を示す挿入位置を特定し、前記添え字変換配列において、前記挿入位置に関して、対応する添え字の範囲を確定するとともに、前記配列の末尾以降の所定の位置を特定するためのオフセット値を与え、前記添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を上方にシフトするとともに、受け入れられた添え字をデクリメントするためのオフセット値を与え、前記配列の末尾に、前記挿入すべき項目値を配置し、前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴とする表形式データの挿入方法により達成される。

【００１１】

本発明によれば、添え字変換配列により、添え字に所定のオフセットを与えて、配列の項目値を特定している。したがって、挿入の場合に、新たに挿入された位置に対応する添え字が、添え字変換配列により、配列の末尾以降の所定の位置に配置された新たな項目値を特定することができる。したがって、実際に配列中に実際にデータを挿入し、他のデータを移動する必要がなく、論理上、配列に項目値を挿入することが可能となる。また、これにより、後述するような並列処理やトランザクション処理を適切になすことが可能となる。

【００１２】

また、本発明の目的は、項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意の位置の項目値を削除する表形式データの削除方法であって、レコード番号を添え字として受け入れ、受け入れられた添え字の範囲に対応するオフセット値を与えるように構成された添え字変換配列を生成し、削除すべき項目値の位置を示す削除位置を特定し、前記添え字変換配列において、削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を下方にシフトするとともに、受け入れられた添え字をインクリメントするためのオフセット値を与え、前記添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値が与えられ、当該オフセット値が与えられた添え字により配列中の項目値が特定されることを特徴

とする表形式データの削除方法により達成される。

本発明においても、添え字変換配列により、削除する位置に対応する項目値を、添え字がポイントしないように構成される。データを実際に削除し、かつ、データを移動することなく、論理上、データ削除の処理が実現される。

また、表形式データの削除方法は、上記更新方法および削除方法を、順次、或いは、その逆順で実行すればよい。

【００１３】

また、本発明の別の実施態様においては、項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データの項目値の値を変換する表形式データの値変換方法は、項目値の範囲に対応するオフセット値を与えるように構成された値変換配列を生成し、レコード番号を添え字として受け入れ、前記配列中の、当該添え字に対応する項目値に、前記項目値の範囲に対応したオフセット値を与えるように構成されている。

また、本発明の目的は、項目とこれに含まれる項目値とを含むレコードの配列として表わされる表形式データにおいて、任意のレコード番号に対応する項目値を特定するように構成された表形式データのデータ構造であって、当該表形式データを、特定の項目に属する項目値に対応した項目値番号の順に当該項目値が格納されている第１の実体配列を含む値リストと、一意的なレコード番号の順に、当該項目値番号を指示するためのポインタ値が格納された第２の実体配列を含むポインタ配列とからなる一以上の情報ブロックに分割するように構成し、前記ポインタ配列に、入力されたレコード番号を添え字として、当該添え字の範囲に対応する第１のオフセットを与えるように構成された第１の添え字変換配列を形成するとともに、当該添え字変換配列を経てオフセットが与えられた添え字により特定される、第２の実体配列中のポインタ値を値として、当該値の範囲に対応する第２のオフセットを与えるように構成された値変換配列を形成し、前記値リストに、入力された、ポインタ配列の出力を添え字として、当該添え字の範囲に対応する第３のオフセットを与えるように構成された第２の添え字変換配列を形成し、当該第２の添え字変換配列を経てオフセットが与えられたポインタ配列の出力により、第１の実体配列中の項目値が特定されるように構成されたことを特徴

とする表形式データのデータ構造により達成される。

【００１４】

本発明によれば、たとえば、あるプロセスでは、添え字変換配列や値変換配列を経たデータストリームを形成し、他のプロセスでは、実体配列のみを利用するデータストリームを形成することができる。このように、必要に応じて、上記変換配列の通過や不通過を設定することにより、プロセスの並列処理やトランザクション処理を適切に実現することが可能となる。

たとえば、前記添え字変換配列の各々は、所定の範囲にそれぞれ含まれる添え字の最小値を示す開始位置からなる開始位置配列、および／または、当該所定の範囲に含まれる添え字の最大値を示す終了位置からなる終了位置配列と、対応するオフセット値からなるオフセット配列とから構成されるのが好ましい。また、前記値変換配列は、所定の範囲にそれぞれ含まれる値の最小値を示す開始位置からなる開始位置配列、および／または、当該所定の範囲に含まれる値の最大値を示す終了位置からなる終了位置配列と、対応するオフセット値からなるオフセット配列とから構成されるのが好ましい。

【００１５】

さらに、本発明の目的は、上に定義したデータ構造にて表わされる表形式データの任意の位置に、項目値を挿入する表形式データの挿入方法であって、

（１）前記値リストに関して、

挿入される項目値の位置を示す挿入位置を特定し、前記第２の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第３のオフセット値を与え、前記第２の添え字変換配列において、前記挿入位置に対応する添え字の値より大きなものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第３のオフセット値を与え、前記第１の実体配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

（２）前記ポインタ配列に関して、

挿入されるレコード番号に対応するポインタ値の位置を示す挿入位置を特定し

、前記第１の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第１のオフセット値を与え、前記第１の添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第１のオフセット値を与え、前記第２の実体配列の末尾以降の所定の位置に、現存するポインタ値より大きな新たなポインタ値を配置し、かつ、

（３）前記ポインタ配列に関して、

前記値変換配列に、前記値リストにおける挿入位置に対応するポインタ値以上を有するものに対して、これをインクリメントする第２のオフセット値を与え、前記値変換配列に、前記新たなポインタ値が、前記挿入位置に対応する位置を特定するような第２のオフセット値を与えることを特徴とする表形式データのデータ挿入方法により達成される。

【００１６】

上記挿入方法の好ましい実施態様においては、ｍ個のレコード番号と、ｎ個の項目値とを備え、前記値リスト中の項目値の挿入位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾ｎに項目値が配置され、かつ、レコード番号の挿入位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾ｍにポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、値がｉである場合に、第３のオフセット値として（ｎ－ｉ）を与え、かつ、値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、添え字がｊである場合に、第１のオフセット値として（ｍ－ｊ）を与え、かつ、添え字が（ｊ＋１）以上ｍ以下の場合に、第１のオフセット値として（－１）を与え、

（３）前記ポインタ配列の値変換配列において、値の範囲が（ｉ－１）の場合に

、第２のオフセット値として０を与え、値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、値がｎである場合に、第２のオフセット値として（ｉ－ｎ）を与えるように構成されている。

【００１７】

或いは、ｍ個のレコード番号と、ｎ個の項目値とを備え、前記値リスト中の項目値の挿入位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾以降の所定の位置ｚ（ｚ≧ｎ）に項目値が配置され、かつ、レコード番号の挿入位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾以降の所定の位置ｘ（ｘ≧ｍ）にポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、

値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、値がｉである場合に、第３のオフセット値として（ｚ－ｉ）を与え、かつ、値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、添え字がｊである場合に、第１のオフセット値として（ｘ－ｊ）を与え、かつ、添え字が（ｊ＋１）以上ｍ以下の場合に、第１のオフセット値として（－１）を与えるとともに、

（３）前記ポインタ配列の値変換配列において、値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、値がｙ（ただし、ｙは、第２の実体配列の位置ｘに格納されたポインタ値）である場合に、第２のオフセット値として（ｉ－ｙ）を与えるように構成されてもよい。

【００１８】

また、本発明の目的は、上記定義に基づくデータ構造にて表わされる表形式データの任意の位置の項目値を削除する表形式データの削除方法であって、前記ポインタ配列に関して、削除されるポインタ値の位置を示す削除位置を特定し、

前記第１の添え字変換配列において、前記削除に関して、対応する添え字を確定し、前記第１の添え字変換配列において、前記削除位置に対応するレコード番

号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第１のオフセット値を与えることを特徴とする表形式データの削除方法により達成される。

上記削除方法の好ましい実施態様においては、ｍ個のレコード番号と、ｎ個の項目値とを備え、レコード番号の削除位置がｊ（０≦ｊ≦ｍ－１）である場合に、前記ポインタ配列の第１の添え字変換配列において、添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、かつ、添え字の範囲がｊ以上（ｍ－２）以下である場合に、オフセット値として１を与えるように構成されている。

【００１９】

さらに、本発明の目的は、上記定義によるデータ構造にて表わされる表形式データの任意の位置の項目値を更新する表形式データの更新方法であって、

（Ａ）（１）前記値リストに関して、

更新される項目値を挿入される項目値と仮定し、当該項目値の挿入すべき位置を示す挿入位置を特定し、前記第２の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するためのオフセット値を与え、前記第２の添え字変換配列において、前記挿入位置に対応する添え字の値より大きなものに関して、対応する添え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第３のオフセット値を与え、前記第１の実体配列の末尾以降の所定の位置に、前記挿入すべき項目値を配置し、

（２）前記ポインタ配列に関して、更新されるポインタ値を挿入されるポインタ値と仮定し、当該ポインタ値の挿入すべき位置を示す挿入位置を特定し、前記第１の添え字変換配列において、前記挿入位置に関して、対応する添え字を確定するとともに、前記第１の実体配列の末尾以降の所定の位置を特定するための第１のオフセット値を与え、前記第１の添え字変換配列において、前記挿入位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添

え字の範囲を、当該範囲を確定する値が大きくなるように上方にシフトするとともに、受け入れられた添え字をデクリメントするための第１のオフセット値を与え、前記第２の実体配列の末尾以降の所定の位置に、現存するポインタ値より大きな新たなポインタ値を配置し、かつ、

（３）前記ポインタ配列に関して、前記値変換配列に、前記値リストにおける挿入位置に対応するポインタ値以上を有するものに対して、これをインクリメントする第２のオフセット値を与え、前記値変換配列に、前記新たなポインタ値が、前記挿入位置に対応する位置を特定するような第２のオフセット値を与え、

(B)前記ポインタ配列に関して、更新すべきポインタ値の位置を、削除位置として、前記挿入位置を考慮して特定し、前記第１の添え字変換配列において、前記削除位置に関して、対応する添え字を確定し、前記第１の添え字変換配列において、前記削除位置に対応するレコード番号より大きなレコード番号を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第１のオフセット値を与え、

前記(A)および(B)、或いは、前記(B)および(A)の何れかを順次実行するように構成されたことを特徴とする表形式データの更新方法により達成される。

【００２０】

上記更新方法の好ましい実施態様においては、ｍ個のレコード番号と、ｎ個の項目値とを備え、前記値リスト中の項目値の更新位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾ｎに項目値が配置され、かつ、レコード番号の更新位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾ｍにポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、値がｉである場合に、第３のオフセット値として（ｎ－ｉ）を与え、かつ、値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、添え字がｊである場

合に、第１のオフセット値として（ｍ－ｊ）を与え、かつ、添え字が（ｊ＋１）以上（ｍ＋１）以下の場合に、第１のオフセット値として０を与え、

（３）前記ポインタ配列の値変換配列において、値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、値がｎである場合に、第２のオフセット値として（ｉ－ｎ）を与えるように構成されている。

【００２１】

或いは、ｍ個のレコード番号と、ｎ個の項目値とを備え、前記値リスト中の項目値の更新位置が、ｉ（０≦ｉ≦ｎ－１）であり、第１の実体配列の末尾以降の所定の位置ｚ（ｚ≧ｎ）に更新された項目値が配置され、かつ、レコード番号の更新位置がｊ（０≦ｊ≦ｍ－１）であり、第２の実体配列の末尾以降の所定の位置ｘ（ｘ≧ｍ）に更新されたポインタ値が配置される場合に、

（１）前記値リストの第２の添え字変換配列において、値の範囲が（ｉ－１）以下の場合に、第３のオフセット値として０を与え、値がｉである場合に、第３のオフセット値として（ｚ－ｉ）を与え、かつ、値の範囲が（ｉ＋１）以上ｎ以下である場合に、第３のオフセット値として（－１）を与え、

（２）前記ポインタ配列の第１の添え字変換配列において、添え字の範囲が（ｊ－１）以下の場合に、第１のオフセット値として０を与え、添え字がｊである場合に、第１のオフセット値として（ｘ－ｊ）を与え、かつ、添え字が（ｊ＋１）以上（ｍ＋１）以下の場合に、第１のオフセット値として０を与えるとともに、

（３）前記ポインタ配列の値変換配列において、値の範囲が（ｉ－１）の場合に、第２のオフセット値として０を与え、値の範囲がｉ以上（ｎ－１）以下である場合に、第２のオフセット値として１を与え、かつ、値がｙ（ただし、ｙは、第２の実体配列の位置ｘに格納されたポインタ値）である場合に、第２のオフセット値として（ｉ－ｙ）を与えるように構成してもよい。

【００２２】

或いは、上記定義にしたがった表形式データの削除方法は、　前記レコード番号に関して、レコード番号自体を配置した第４の実体配列と、当該レコード番号を特定するための添え字の範囲にしたがって、所定のオフセット値を与える第３

の添え字変換配列とを設け、前記削除すべきレコード番号の位置を示す削除位置を特定し、前記第３の添え字変換配列において、前記削除に関して対応する添え字を確定し、前記第３の添え字変換配列において、前記削除位置に対応する添え字より大きな値を有するものに関して、対応する添え字の範囲を、当該範囲を確定する値が小さくなるように下方にシフトするとともに、受け入れられた添え字をインクリメントするための第４のオフセット値を与えるように構成してもよい。

これは、レコード番号自体を配置した第４の実体配列を含む情報ブロックを形成し、レコード番号を指定するための添え字変換配列を形成し、これにより実質的なデータの削除を実現している。

【００２３】

また、本発明の目的は、上記方法により表形式データにおけるデータの挿入、削除および／または更新をなし、ロールバックおよびコミットの何れかの場合には、前記添え字変換配列および値変換配列を廃棄することを特徴とするトランザクション処理方法によっても達成される。この場合に、ロールバックの場合にのみ、前記添え字変換配列および値変換配列を廃棄してもよい。

また、本発明の目的は、上記定義にしたがったデータ構造を有する表形式データに関して、複数のプロセスにより同時に処理が実行されるプロセスの並列処理方法であって、前記プロセスのうち、データの挿入、削除および／または更新を伴うものは、前記ポインタ配列において、第１の添え字変換配列、第２の実体配列および値変換配列を経るように構成されるとともに、前記値リストにおいて、第２の添え字変換配列および第１の実体配列を経るように構成され、その一方、前記プロセスのうち、データの挿入、削除および／または更新を伴わないものは、前記ポインタ配列において、第２の実体配列を経るとともに、前記値リストにおいて、前記値リストにおいて、第１の実体配列を経るように構成されたことを特徴とするプロセスの並列処理方法によっても達成される。

データの挿入、削除および／または更新を伴うプロセスは、上述したような挿入方法、削除方法および更新方法の何れかを利用しているのが好ましい。

【００２４】

或いは、本発明のさらに別の実施態様においては、上記データ構造を有する表形式データに関して、レコードのロックをなす方法は、ロックを管理する配列を含む情報ブロックを設け、前記情報ブロックの配列中に、前記レコード番号の各々に対応して、ロックの種別を示す項目値を配置するように構成されている。

また、本発明の目的は、上記種々の方法の少なくとも何れかを実現するプログラムを記憶した、コンピュータにより読み出し可能な記憶媒体によっても達成される。

【００２５】

【発明の実施の形態】

以下、添付図面を参照して、本発明の実施の形態につき説明を加える。図１は、本発明の実施の形態にかかる表形式データにかかる、ある項目のデータを挿入、削除および更新する方法を実現できるコンピュータシステムのハードウェア構成を示すブロックダイヤグラムである。図１に示すように、このコンピュータシステム１０は、通常のものと同様の構成であり、プログラムを実行することにより、システム全体および個々の構成部分を制御するＣＰＵ１２、ワークデータなどを記憶するＲＡＭ(Random Access Memory)１４、プログラム等を記憶するＲＯＭ(Read Only Memory)１６、ハードディスク等の固定記憶媒体１８、ＣＤ－ＲＯＭ１９をアクセスするためのＣＤ－ＲＯＭドライバ２０、ＣＤ－ＲＯＭドライバ２０や外部ネットワーク（図示せず）と接続された外部端子との間に設けられたインタフェース（Ｉ／Ｆ）２２、キーボードやマウスからなる入力装置２４、ＣＲＴ表示装置２６を備えている。ＣＰＵ１２、ＲＡＭ１４、ＲＯＭ１６、外部記憶媒体１８、Ｉ／Ｆ２２、入力装置２４および表示装置２６は、バス２８を介して相互に接続されている。

【００２６】

本実施の形態にかかる表形式データにかかるある項目のデータを挿入するプログラム、ある項目のデータを削除するプログラム、ある項目のデータを更新するプログラム、トランザクション処理を実現するプログラム、並列処理を司るプログラム、および、排他処理を実現するプログラムは、ＣＤ－ＲＯＭ１９に収容され、ＣＤ－ＲＯＭドライバ２０に読取られても良いし、ＲＯＭ１６に予め記憶さ

れていても良い。また、いったんＣＤ－ＲＯＭ１９から読み出したものを、外部記憶媒体１８の所定の領域に記憶しておいても良い。或いは、上記プログラムは、ネットワーク（図示せず）、外部端子およびＩ／Ｆ２２を経て外部から供給されるものであっても良い。なお、本明細書において、データの挿入とは、隣接するレコードの間にデータを入れることのみならず、レコードの末尾に新規のレコードを追加することも含む。

【００２７】

また、本実施の形態においては、データの挿入、削除および更新処理、並びに、トランザクション処理、並列処理、排他処理を適切かつ高速に実現するために、後述するように、表形式データに基づき、所定のデータ形式の情報ブロックを生成する必要がある。この情報ブロック生成プログラムも同様に、ＣＤ－ＲＯＭ１９に収容され、ＲＯＭ１６に記憶され、或いは、外部記憶媒体１８に記憶されても良い。或いは、これらプログラムは、ネットワーク（図示せず）を介して、外部から供給されても良いことはいうまでもない。また、本実施の形態において、情報ブロック生成プログラムにて生成されたデータ（情報ブロック）は、ＲＡＭ１４に記憶され、或いは、外部記憶媒体１８の所定の領域に記憶される。

【００２８】

まず、本発明の前提となる表形式データの取扱い、すなわち、表形式データを管理する形態につき簡単に説明を加える。本発明者は、表形式データを高速に検索、集計およびソートを高速に実現するため、また、複数の表形式データを所望のようにジョインできるようにするため、特定のデータ形式を有する表形式データの構築（特願平１０－２２７２７８号）と、当該特定のデータ形式を有する表形式データのジョイン（特願平１１－１５１１５６号）とを考案している。本発明においても、基本的には、これら出願に開示された技術に基づき、表形式データを所定の情報ブロックの集合体として構築し、これを用いて検索、集計およびソートを実現している。

図２は、本実施の形態にて用いる情報ブロックを示す図である。図２に示すように、情報ブロック１００は、値リスト１１０と値リストへのポインタ配列１２０とを含んでいる。値リスト１１０は、表形式データの各項目に対して、その項

目に属する項目値が順序付け（整数化）された項目値番号の順番に、上記項目値番号に対応した項目値１１１が格納されたテーブルである。値リストへのポインタ配列１２０は、表形式データのある列（すなわち項目）の項目値番号、つまり値リスト１１０へのポインタが表形式データのレコード番号順に格納された配列である。

【００２９】

上記値リストへのポインタ配列１２０と値リスト１１０とを組み合わせることにより、あるレコード番号が与えられたときに、所定の項目に関する値リストへのポインタ配列１２０からそのレコード番号に対応して格納された項目値番号を取り出し、次いで、値リスト１１０内でその項目値番号に対応して格納された項目値を取り出すことにより、レコード番号から項目値を得ることができる。したがって、従来のデータ表と同様に、レコード番号（行）と項目（列）という座標を用いてすべてのデータ（項目値）を参照することができる。

たとえば、図３（ａ）に示す表形式データを考える。この例では、顧客ＩＤ、顧客名、電話番号という項目に種々の項目値が与えられている。本実施の形態においては、このような表形式データを、図３（ｂ）ないし（ｄ）に示す形式の情報ブロックとして保持している。たとえば、図３（ｂ）において、ポインタ配列１２０－１は、顧客ＩＤを示す項目値を格納した値リスト１１０－１に関連付けられている。すなわち、先頭レコード（レコード番号「０」）のポインタ配列のポインタ値は０であり、これに対応して、顧客ＩＤを示す項目値「１」が得られる。図３（ｃ）において、ポインタ配列１２０－２は、顧客名を示す項目値を格納した値リスト１１０－２に関連付けられている。たとえば先頭レコード（レコード番号「０」）のポインタ配列におけるポインタ値は「５」であり、これに対応して、顧客名を示す項目値「山田　○男」が得られる。図３（ｄ）においても、同様に、ポインタ配列１２０－３が、電話番号を示す項目値を格納した値リスト１１０－３に関連付けられていることが理解できよう。また、各値リストにおいては、項目値が順序付けられて（この例では昇順）いることが理解できよう。

【００３０】

さらに、本実施の形態においては、情報ブロック１００の値管理テーブルは、

値リスト１１０のほか、検索や集計のために用いる分類番号フラグ配列、項目値に対応するポインタを格納すべきメモリ空間の先頭アドレスを示す開始位置配列、および、存在数配列が含まれている。分類番号フラグ配列の各フラグ、および、存在数配列の各存在数は、項目値の各々に対応付けられている。分類番号フラグのフラグ値は、通常「０」であり、検索や集計の際に見出すべき項目値に対応して「１」にセットされる。また、存在数は、その項目値を有するレコードの個数に対応する。なお、開始位置は、対応するポインタ値よりも小さなポインタ値に対応する存在数を加算したものに対応するため、必ずしも設ける必要はない。

図４（ａ）は、表形式データの他の例を示す図、図４（ｂ）および（ｃ）は、それぞれ、「性別」および「年令」に関する情報ブロックを示す図である。図４（ｂ）に示すように、性別に関する情報ブロック２００－１の値管理テーブル２１０－１には、ポインタ配列２２０の各ポインタ値に対応する項目値（「男性」および「女性」）と、各項目値に対応する分類番号、開始位置および存在数が示されている。たとえば、ポインタ値が「０」（つまり、値リストの項目値が「男性」）であるようなレコードの数は６３２５６４個であり、その一方、ポインタ値が「１」（つまり、値リストの項目値が「女性」）であるようなレコードの数は３６７４３６個となっている。また、各項目値に対応する開始位置は、後述するレコードへのポインタ配列２３０－１の先頭アドレスを示している。図４（ｃ）においても、同様のことが理解できよう。

【００３１】

このようなデータ構造を有する情報ブロックを用いた検索の一例および情報ブロックの生成処理につき、以下に説明する。図５は、単一項目に関する検索手法を示すフローチャートである。この処理は、ＣＰＵ１２（図１参照）が所定の検索プログラムを実行することにより実現される。この例では、「年令」の項目値が１６歳または１９歳であるレコードが検索される。まず、表形式データに関する情報ブロックのうち、図４（ｃ）に示す「年令」に関する情報ブロック２００－２が特定される（ステップ５０１）。

【００３２】

次いで、特定された情報ブロック（以下、「特定情報ブロック」と称する。）の値リスト２１０－２において、項目値が上記検索条件に合致するもの（１６歳または１９歳）に対応する行の分類番号が「１」にセットされる（ステップ５０２）。本例の場合には、項目値番号「０」および項目値番号「３」に対応する行の分類番号が１にセットされる。次いで、分類番号が「１」にセットされている行に対応した開始位置および存在数が取得される（ステップ５０３）。これら情報をポインタ取り出し情報と称する。レコードへのポインタ配列において、ステップ５０３にて取得されたポインタ取り出し情報に基づき、検索条件に合致したレコードへのポインタを示すレコード番号が取り出される（ステップ５０４）。本例においては、項目値番号「０」に対応したレコードのポインタは、レコードへのポインタ配列の開始位置「０」すなわち先頭から、４５８９８個目までの領域に格納され、その一方、項目値番号「３」に対応したレコードのポインタは、レコードへのポインタ配列の２３８３１３７番目から１８９６５３個分の領域に格納されていることがわかる。最後に、後の処理にて利用できるようにするために、取り出されたレコード番号の配列が、結果集合として作成され、これが保持される（ステップ５０５）。

また、集計およびソートも、分類番号、開始位置および存在数を利用することにより実現することができる。

【００３３】

次に、上述したような検索処理等に利用するための情報ブロックの生成処理につき説明を加える。図６は、表形式データに基づき情報ブロックを作成するための処理を説明するフローチャートである。まず、システム１０は、表形式の原データを取得し、これを項目別のものに分解する（ステップ６０１）。この原データは、たとえば、図７（ａ）に示すものでも良いし、或いは、図７（ｂ）に示すものでも良い。これら原データは、外部から供給されるものであっても良いし、或いは、固定記憶媒体１８に記憶されたものであっても良い。以下に述べるステップ６０２ないしステップ６０４からなる処理ブロック６１０は、ある一つの項目に関する情報ブロックの生成を示す。したがって、複数の項目に関する情報ブロックを生成する場合には、項目の数だけ処理ブロック６１０に対応する処理が

実行される。以下、「性別」に関する項目の情報ブロックを例にとって説明を加える。

【００３４】

まず、「性別」に関する項目の情報ブロック用の領域が、たとえば、ＲＡＭ１４中に確保される（ステップ６０２）。次いで、この確保された領域中に、値管理テーブルが生成される。より詳細には、まず、値管理テーブルが初期化される。次いで、原データのうち、「性別」に関するデータを先頭から末尾まで走査することにより、どのような項目名が、それぞれいくつ存在するかが見出される。本例では、「女性」および「男性」という項目名が、それぞれ、３６７４３６個および６３２５６４個だけあることが見出される。これにより、値リストに、「女性」および「男性」という項目値がセットされ、また、存在数配列にも所定の数がセットされる。その後に、項目値が所定の基準にしたがってソートされる。ソートの際には、項目数の並び替えにしたがって存在数も並びかえられる。次いで、開始位置配列の値が決定される。これは、ソートにより自己より上位に位置する存在数を累算することにより求められる。また、開始位置配列の値を、対応する分類番号配列の値に割り当てる。この値は次のステップにて用いられる。このようにして値管理テーブルが生成された後に、レコードへのポインタ配列が生成される。このポインタ配列の領域の大きさは、存在数の総和に対応する。

このようにして、所定の項目に関する情報ブロックを作り出すことが可能となる。この情報ブロックの生成を予め行っておき、生成された情報ブロックを用いて検索、集計およびソートの処理が実行される。

【００３５】

次に、上記形態の情報ブロックにおいて、要素を挿入、削除或いは更新する処理につき、以下に説明を加える。

たとえば、１００万レコード（レコード番号０～９９９，９９９）のサイズを持つ配列の先頭から２番目に、１つの要素を挿入することを考える。たとえば、レコード番号０～９９９，９９９に対応する配列のうち、レコード番号「０」と「１」との間に、新たに一つの要素を挿入する場合である（図８（ａ）参照）。この場合に、新たなレコード番号「１」に新たな値（たとえば「１０」が挿入さ

れ）、これ以降の値（すなわち９９９，９９９個の値）を一つずつ移動しなければならない。また、特定の要素を削除する場合にも、莫大な数の要素を移動する必要がある。或いは、上記配列において、特定の値を持つ要素を別の値に書きかえる場合、たとえば、１以上の値を有するものをインクリメントする場合（図８（ｂ）参照）にも、配列全体を走査する必要が有り、処理に莫大な時間を要する。

ところで、上記配列は、添え字が配列中の位置を示すことにより、対応する位置の格納値を取り出す（すなわち添え字から格納値を引用する）ことができる点で有利である。つまり、各要素（格納値）がメモリのアドレス順に格納されていれば、添え字から直ちにアドレスを算出し、対応するアドレスの拡張値を得ることができる。そこで、この利点を生かしつつ、要素（格納値）の挿入および削除に要する負荷を最小にする手法が求められる。

【００３６】

本発明においては、図９（ａ）に示すような、添え字を配列に与え、対応する値（格納値）を取り出すような系において、図９（ｂ）ないし（ｄ）に示すように、配列に与える「添え字」を変換する添え字変換部、および／または、配列から取り出された値を変換する値変換部を設けるとともに、配列自体を、実体配列と差分配列とに分割し、配列に含まれる要素（格納値）の位置を変更することなく、かつ、値自体を変更することなく、要素の挿入、削除および更新を実現可能にしている。

【００３７】

[添え字変換配列による要素の挿入]

まず、本発明の第１の実施の形態にて利用される、要素の挿入等のための添え字変換配列につき、説明を加える。図２ないし図４に示す情報ブロックに関して考えると、添え字には、値リストへのポインタ配列の要素（ポインタ値）を指定するためのレコード番号や、値リストの項目値を指定するためのポインタ配列中のポインタ値等が含まれる。また、配列には、値リストへのポインタ配列、値リスト、レコード番号へのポインタ配列等が含まれる。

たとえば、図１０（ａ）に示すように、配列の第０行（添え字「０」）と第１

行（添え字「１」）との間に、要素「Ｙ０」を挿入する場合を考える。この場合には、要素「Ｙ０」の挿入後には、添え字と配列中の要素（格納値）とは、図１０（ｂ）に示す論理的関係となる。

【００３８】

図１０（ｂ）に示す論理的関係を保持しつつ、挿入時に生じる負荷を最小限にするために、図１０（ｃ）に示す添え字変換配列を考える。図１０（ｃ）において、添え字変換配列は、開始位置配列と、終了位置配列と、オフセット配列とからなる。これらの配列に格納された要素（値）のうち、同じ位置（たとえば第０行、第１行等）に配置されたものが相互に関連している。開始位置配列中の要素と、終了位置配列中の対応する要素により、関連する添え字の範囲が特定される。すなわち、開始位置配列中の要素は、関連する添え字の最小値を示し、終了位置配列中の要素は、関連する添え字の最大値を示す。その一方、オフセット配列中の対応する要素は、添え字に加減算して、実際の実体配列を指定する変換済の添え字を得るためのオフセット値を示す。なお、本明細書において、開始位置配列、終了位置配列およびオフセット配列の対応する要素の組（すなわち同じ行に位置する要素の組）を、場合によって「構造体配列」と称し、各配列中の要素を「構造体のメンバー」と称する。

この添え字変換配列の意味するところにつき以下に説明する。上述したように、ある構造体配列中の構造体のメンバーにより、添え字の範囲および当該範囲に含まれる添え字のオフセット値が決定される。このため、ある添え字に対して、その添え字に関連する構造体配列を特定し、オフセット値を加えることにより、変換済の添え字を得ることができ、これを使用して配列（実体配列）の位置を特定して、配列中の要素（格納値）を参照することが可能となる。

【００３９】

たとえば、図１０（ｃ）において、添え字「０」は、第０行（１行目）の構造体配列と関連することが理解できる。この構造体配列において、オフセット配列中のメンバーの値（オフセット値）は「０」であるため、変換済の添え字は、「０＋０」＝０となる。したがって、もともとの添え字「０」に対応する実体配列中の要素は、「０」の位置に格納されていることが分かる。また、添え字「１」

は、第１行（２行目）の構造体配列と関連し、当該構造体配列に関するオフセット配列中のメンバーの値（オフセット値）は、「ｎ－１」であるため、変換後の添え字は、「１＋（ｎ－１）＝ｎ」となる。したがって、対応する要素は、実体配列の「ｎ」の位置に格納されていることが理解できる。添え字「２」ないし「ｎ」に関しても同様に、対応する要素の格納されている位置が特定できることは明らかであろう。

【００４０】

より詳細に、上記手順につき、図１１のフローチャートを参照して説明を加える。まず、添え字の位置を示す番号（格納位置番号）が初期化され（ステップ１１０１）、次いで、開始配列および終了配列の要素を参照して、添え字が、開始位置配列および終了位置配列の値により画定される範囲のうち、何れに含まれるかが特定される（ステップ１１０２）。すなわち、このステップにより、添え字がどの構造体配列に関連するかが特定される。たとえば、図１０（ｃ）の例において、添え字「０」は、開始配列中の要素「０」および終了配列の要素「０」にて特定される範囲に含まれることが分かる。或いは、添え字「１」は、開始配列中の要素「１」および終了配列中の要素「１」にて特定される範囲に含まれることがわかる。

【００４１】

次いで、その範囲に基づき、対応するオフセット値が取り出され（ステップ１１０３）、添え字にオフセット値を加えた値が得られる（ステップ１１０４）。このオフセットが加えられた値が、変換済の添え字となるため、この値を利用して、実体配列中の要素が特定される（ステップ１１０５）。たとえば、添え字「１」に関して、この添え字は第１行（２行目）に対応する範囲に含まれるため、第１行のオフセット値である「ｎ－１」が添え字「１」に加えられる。したがって、変換済の添え字は「ｎ」となり、実体配列中の対応する要素（格納値）「Ｙ０」を得ることができる。或いは、添え字「２」に関して、この添え字は、第２行（３行目）に対応する範囲に含まれるため、第２行のオフセット値である「－１」が添え字「２」に加えられる。しがたって、変換済の添え字は「１」となり、実体配列中の対応する要素（格納値）「Ｘ１」を得ることができる。

必要な添え字につきステップ１１０２ないし１１０５の処理を繰り返すことにより（ステップ１１０６および１１０７参照）、図１０（ｂ）に示す論理的関係を維持しつつ、実体配列中の要素（格納値）を得ることができる。したがって、本発明においては、配列中の要素（格納値）の挿入時には、上述したような変換配列を生成するとともに、配列中の要素自体を移動させることなく、実体配列の所定の位置（たとえば、アドレスの最大値に対応する位置）に、挿入すべき要素を配置することにより、負荷が生じることなく、要素の挿入を実現できるようにしている。

【００４２】

［要素挿入のための添え字変換配列の生成］

次に、第１の実施の形態にかかる上記添え字変換配列の作成手順につき説明を加える。たとえば、図１２に示すように、要素数が５個の配列（符号１２０１）を考え、これのうち、第０行（１行目）と第１行（２行目）との間に、新たな要素（格納値）「Ｙ０」を挿入する場合を考える（符号１２０２参照）。図１３は、配列に新たな要素（格納値）を挿入するための変換配列の生成処理を説明するフローチャートである。この処理は、図１におけるＣＰＵ１２により実行される。まず、初期状態の添え字変換配列が生成される（ステップ１３００）。より詳細には、配列の要素数が調べられ、開始位置および終了位置が画定される。たとえば、図１２に示す例では、要素数が「５」であるため、初期的な添え字変換配列において、開始位置が「０」であり、終了位置が「４」となる。また、この例では、添え字から直接、配列中の要素が参照されるようになっているため、初期的なオフセット値が「０」となる（図１４の符号１４０１参照）。なお、添え字変換配列が予め生成されている場合、すなわち、既に一度以上の挿入がなされている場合や、情報ブロック生成時に初期的な添え字変換配列が生成されている場合には、ステップ１３００は省略される。

【００４３】

次いで、挿入すべき要素の位置が特定され（ステップ１３０１）、挿入位置、挿入位置の前および挿入位置の後に関する添え字変換配列のための領域が確保される。図１４の符号１４０１の例では、初期的な添え字変換配列に含まれる各配

列が、それぞれ１つの要素のみを含んでいたが、ステップ１３０１の処理により、３つの要素（挿入位置の前に関する要素、挿入位置に関する要素、および、挿入位置の後に関する要素）を有する配列のための領域が確保される（ステップ１３０２）。なお、既に各々が複数の要素を有する配列が設けられていた場合には、要素数が２つ増えるような配列のための領域が確保される。

【００４４】

その後、挿入位置より前に関して、各配列の要素の値が決定される（ステップ１３０３）。より詳細には、開始位置配列の対応する要素値およびオフセット配列の対応する要素値は変更されず、終了位置配列の要素値として、「挿入位置に対応する値－１」が与えられる。また、挿入位置に関しても、各配列の対応する要素の値が決定される（ステップ１３０４、１３０５）。この場合には、開始位置配列および終了位置配列の要素値として、「挿入位置に対応する値」が与えられ、かつ、オフセット配列の要素値として、「実体配列の末尾の位置に対応する値」が与えられる。なお、後述するように「差分配列」として、挿入すべき要素を別個の位置に配置するのであれば、当該差分配列中の、挿入すべき要素の配置される位置に対応する値をオフセット値として与えれば良い。

また、挿入位置より後に関しても、以下の手法により各配列の要素値が決定される。すなわち、開始位置配列の対応する要素値として、「挿入位置に対応する値＋１」が与えられ（ステップ１３０６）、終了位置配列の対応する値として、初期的な「終了位置に対応する値＋１」が与えられる（ステップ１３０７）。次いで、オフセット配列の対応する要素値が、「その初期値（元の値）－１」に設定される（ステップ１３０８）。

【００４５】

たとえば、図１２の符号１２０１に示すような添え字および配列があり、ステップ１３００により、図１４の符号１４０１に示す初期的な添え字変換配列が生成された場合には、上記ステップ１３０１ないしステップ１３０７の処理により、符号１４１１に示す添え字変換配列が生成されることが理解できよう。

このような添え字変換配列を生成した後に、ＣＰＵ１２は、ステップ１３０４にて設定された実体配列の末尾以降の所定の位置（上記例では末尾）に、挿入す

べき要素を格納する。これにより、実体配列の要素（格納値）を移動することなく、実質的に（論理上）実体配列の所望の位置に要素を挿入することが可能となる。

【００４６】

[添え字変換配列による要素の削除]

次に、上述した添え字変換配列を用いて、要素を削除する手法につき説明を加える。たとえば、図１５（ａ）に示すように、配列の第１行（２行目）の要素を削除する場合を考える。この場合には、削除の後に、添え字と配列中の要素（格納値）とは、図１５（ｂ）に示す論理的関係となる。挿入の場合と同様に、図１５（ｂ）に示す論理的関係を保持しつつ、削除時に生じる負荷を最小限にするために、図１５（ｃ）に示す添え字変換配列を考える。図１５（ｃ）においても、図１０（ｃ）と同様に、構造体配列により添え字の範囲およびオフセット値が規定され、変換済の添え字が求められるようになっている。より詳細には、添え字の各々に関して、図１１の処理を施すことにより、実体配列中の要素（格納値）が特定されることが理解できる。たとえば、添え字「１」は、第１行（２行目）の構造体配列と関連し、当該構造体配列に関するオフセット配列中のメンバーの値（オフセット値）は「１」であるため、変換後の添え字は、「１＋１＝２」となる。したがって、対応する要素は、実体配列の「２」に対応する位置に格納されていることが理解できる。

【００４７】

[要素削除のための添え字変換配列の生成]

たとえば、図１６（ａ）に示すように、５個の要素からなる配列（符号１６０１）を考え、これのうち、第１行（２行目）の要素（格納値）Ｘ０を削除する場合を考える（符号１６０２参照）。なお、この例では、実体配列は、配列１６０１と同じであると仮定する。

図１７は、配列から要素（格納値）を削除するための変換配列の生成処理を示すフローチャート、図１８（ａ）は、図１６（ａ）に示す例に関して、処理に伴って変換配列中の数値が変化する状態を示す図である。

図１７に示す処理も、図１におけるＣＰＵ１２により実行される。まず、添え

字変換配列が生成されていない場合には、初期状態の添え字変換配列が生成される（ステップ１７００）。たとえば、図１６（ａ）に示す例では、要素数が「５」であるため、その開始位置を示す値が「０」、終了位置を示す値が「４」、オフセット値が「０」となるような開始位置配列、終了位置配列およびオフセット配列が生成される（図１８（ａ）の符号１８０１参照）。

【００４８】

次いで、削除すべき要素の位置が特定され（ステップ１７０１）、当該削除位置に関する変換配列のための領域が特定される（ステップ１７０２）。図１８（ａ）に示す例では、符号１８０２に示すように、変換配列中の各配列において、削除位置の前、削除位置および削除位置の後に関する要素を収容するための領域が選られる。新たな領域において、削除位置の前に関して、終了位置配列の要素として、「削除位置を示す値－１」が配置され（ステップ１７０３）、その一方、削除位置の後に関して、開始位置配列の要素として、「削除位置を示す値＋１」が配置される（ステップ１７０４）。このようにして、削除位置を含む変換配列中の構造体配列に関して、符号１８０２に示すような開始位置配列、終了位置配列およびオフセット配列が作成される。なお、削除位置に関する構造体配列は、最終的には使用されないため、削除位置自体を算出することは必要であるが、図１８（ａ）に示すように、要素を配列中に格納する必要はない。

【００４９】

次いで、各配列に関して、削除位置より後の要素に関して、以下に述べる演算が施される。まず、開始位置配列の要素が「もとの値－１」に設定され（ステップ１７０５）るとともに、終了位置配列の要素が「もとの値－１」に設定される（ステップ１７０６）。また、オフセット配列の要素が「もとの値＋１」と設定される（ステップ１７０７）。たとえば、図１８（ａ）の例において、削除位置より後ろに位置する開始位置配列中の要素「２」がデクリメントされ、「１」となり（符号１８０４参照）、終了位置配列中の要素「４」がデクリメントされ「３」となり、かつ、オフセット配列中の要素「０」がインクリメントされ「１」となる。このようにして、配列中の要素を削除するための添え字変換配列を得ることが可能となる。なお、実体配列（すなわち、添え字にて特定される要素を実

際に格納した配列）は、要素の削除の際にも変更されない。すなわち、添え字変換配列のみによって、見かけ（論理上）の配列中の要素の削除を実現している。

【００５０】

また、図１６（ｂ）に示す他の例についても、図１７の処理により配列中の要素の削除が実現できる。この例では、先に説明した要素の挿入により、配列には、第３行（４行目）に要素「Ｙ０」が挿入されており、したがって、その実体配列は、末尾に要素「Ｙ０」が付加された形態となっている（図１６の符号１６１０参照）。また、上記挿入の際に、図１８（ｂ）の符号１８１１にて示す添え字変換配列が生成されている。このような場合には、ステップ１７００に示す初期変換配列の生成は省略される。図１８（ｂ）の符号１８１１にて示す添え字変換配列に対して、ステップ１７０１ないしステップ１７０７の処理を施すことにより、符号１８１３にて示す添え字変換配列が求められることが理解できよう。また、このような添え字変換配列を用いることで、実体配列（図１６の符号１６１０）を変更することなく、添え字から変換配列を経た添え字（変換済の添え字）を用いて、図１６の符号１６１２に示す論理関係を持たせることができることが理解されよう。

【００５１】

[要素の挿入および削除の結合]

配列中の要素の挿入および削除は、上記図１３に示す挿入処理および図１７に示す削除処理を順次実行することにより実現される。上記処理の順序は逆でも容易なことは言うまでもない。たとえば、初期的に、図１９に示すように、「０」ないし「４」の添え字に対して、それぞれ、「０」、「１００」、「２００」、「３００」および「４００」という要素が特定されるようになっている配列１９０１を考える。この配列に関して、第２行（３行目）と第３行（４行目）との間に、要素「２０１」を挿入するとともに、第１行（２行目）の要素を削除して、符号１９０２に示す配列を得ることを考える。

本実施の形態においては、図２０（ａ）に示すように、まず、初期的な変換配列を作成し（符号２００１参照）、次いで、図１３に示す処理によって挿入のための変換配列（図２０（ｂ）の符号２００２参照）を作成するとともに、実体配

列の末尾に挿入すべき要素（この場合には「２０１」）を配置する（符号２００３参照）。次いで、図１７に示す処理によって削除のための変換配列（図２０（ｃ）の符号２００４参照）を作成する。これにより、要素の挿入および削除が実現される。

【００５２】

なお、要素の削除、および、当該削除した要素の位置に新たな要素を挿入する処理を順次実行することにより、要素の更新が実現されることが理解できるであろう。図２０（ａ）～（ｃ）の右側の図表（符号２０１１～２０１３）において、格納位置は、添え字変換配列を経た添え字、格納値は、見かけ（論理的な）実体配列を示している。初期的には、添え字と格納位置とは一致している（符号２０１１参照）が、挿入および削除により、添え字変換配列が生成ないし更新され、見かけの実体配列において所望のように要素が挿入および削除されていることが理解できよう（符号２０１２、２０１３参照）。

【００５３】

［添え字変換配列の他の例］

次に、第２の実施の形態にかかる添え字変換配列のサイズを削減した例につき説明を加える。前記第１の実施の形態においては、開始位置配列と終了位置配列とを用いて、添え字がどの範囲に含まれるか、或いは、削除または挿入すべき要素の位置が特定されるようになっていた。しかしながら、終了位置配列のある要素の値が「ｎ」である場合に、次の行に位置する開始位置配列の要素の値は、「ｎ＋１」であることは明らかであり、その逆もまた明らかである。そこで、第２の実施の形態においては、添え字変換配列を、開始位置配列およびオフセット配列、或いは、終了位置配列およびオフセット配列から構成し、添え字変換配列に要するメモリサイズを削減した。

たとえば、図２１（ａ）に示す添え字変換配列２１０１を、図２１（ｂ）のように表わしても良い。図２１（ｂ）においては、０≦添え字ｉ≦（３－１）であるような添え字「ｉ」は、第０行（１行目）に対応するため、このような添え字のオフセット値は「０」となる。また、３≦添え字ｉ≦（４－１）であるような添え字「ｉ」（＝３）は、第１行（２行目）に対応し、４≦添え字ｉであるよう

な添え字「ｉ」は、第２行（３行目）に対応する。

【００５４】

或いは、図２１（ｃ）に示すような添え字変換配列を用いても良いことは明らかである。この場合に、５≧添え字ｉ≧（３＋１）であるような添え字「ｉ」は、第２行（３行目）に対応し、３≧添え字ｉ≧（２＋１）であるような添え字「ｉ」は、第１行（２行目）に対応し、また、２≧添え字ｉであるような添え字「ｉ」は、第０行（１行目）に対応する。

なお、図２１（ｂ）に示すような添え字変換配列を用いれば添え字の総数を知ることができ、これにより、配列のサイズを知ることができるため、この添え字変換配列を用いるのがより好ましい。

【００５５】

[実体配列と差分配列]

前記第１の実施の形態においては、配列に要素を挿入する際に、配列の末尾に挿入すべき要素を配置している（たとえば、図１０（ｃ）参照）。しかしながら、オフセット配列中の対応するオフセット値を設定することにより、所望の位置に、挿入すべき要素を配置できることは明らかである。第３の実施の形態では、この挿入される要素を、任意の編集用作業領域に配置している。このような、挿入される要素が配置された編集用の作業領域を「差分配列」と称する。

図２２は、添え字変換配列、実体配列および差分配列の一例を示す図である。図２２に示すように、この例では、添え字変換配列２２０１により、挿入された要素「Ｙ０」および「Ｙ１」が、実体配列２２０２とは別個の領域に確保された差分配列２２０３中に配置されている。これは、添え字変換配列を作成（図１３参照）した後に、挿入すべき要素（格納値）を配置する際に、挿入すべき要素のオフセット値にしたがった領域に、差分配列を作成し、当該差分配列中に上記要素を配置すれば良い。

【００５６】

[値変換配列]

次に、本発明の第４の実施の形態にかかる値変換配列につき説明を加える。配列に関しては、上記要素の挿入および削除のほか、配列の要素（格納値）自体を

変化させる（更新する）必要が生じ得る。本実施の形態においては、配列の要素を更新するために、以下に述べる値変換配列を利用する。

たとえば、図２３（ａ）に示すように、ある配列（レコード番号リスト）のうち、１以上の値を有するものをインクリメントする必要がある場合を考える。従来の手法では、配列中の要素（格納値）が１以上であるかを判断し、１以上である場合には、要素（格納値）をインクリメントするという処理を、全ての要素に関して実行していた。これに対して、本発明の第４の実施の形態においては、配列の後段（下流側）に、値変換配列を配置し、配列から出力された値を、値変換配列を通すことにより、変換済の値を出力し、この変換済の出力において、要素（格納値）の更新が実現されているような手法を用いている。

図２３（ａ）に示すような場合には、図２３（ｂ）に示すような値変換配列が設けられる。この実施の形態では、第２の実施の形態と同様に、値変換配列は、終了位置配列とオフセット配列とから構成されている。無論、第１の実施の形態のように、値変換配列を開始位置配列、終了位置配列およびオフセット配列から構成しても良い。

【００５７】

これらの配列に格納された要素（値）のうち、同じ位置（たとえば第０行、第１行）に配置されたものが相互に関連している。したがって、対応する位置に配置された要素により、構造体配列を構成する。本実施の形態においては、終了位置配列の値により、実体配列中の要素（格納値）の配置されている位置が、どの構造体配列に関連しているかが特定され、それに基づき、値に対するオフセットを特定することが可能となる。上述した例（１以上の要素をインクリメントする例）に関する処理につき、図２４を用いてより詳細に説明する。この処理も、先に延べた実施の形態と同様に、図１に示すＣＰＵにより実行される。

【００５８】

まず、要素の格納位置を示す番号（格納位置番号）が初期化され（ステップ２４０１）、当該格納位置番号により特定される配列中の要素（格納値）が、終了位置配列により画定される範囲のうち、何れに含まれるかが特定される（ステップ２４０２）。このステップにより、要素がどの構造体配列に関連するかが特定

される。図２３（ｂ）の例において、格納位置番号が「０」（すなわち、第０行（１行目）の）要素は「１」であり、この要素は、終了位置配列中の第１行（２行目）の値にて確定される範囲（すなわち、１≦配列中の要素≦１）に含まれるため、当該要素は、第１行（２行目）の構造体配列に関連していることがわかる。

次いで、この構造体配列に属するオフセット配列中のオフセット値が取り出され（ステップ２４０３）、要素にオフセット値を加えた値が出力される（ステップ２４０４）。このように、値変換配列を経た値は、出力値配列の格納位置番号に対応する領域に記憶されても良い（ステップ２４０５）。或いは、値変換配列を経た値を直接他の処理に利用しても良いことは言うまでもない。上記格納位置番号が「０」の要素に関しては、対応するオフセット値は「１」であるため、「１＋１＝２」が変換配列後の要素（格納値）として出力される。

実体配列中の必要な値に関して、ステップ２４０２ないし２４０５の処理を、繰り返すことにより、図２３（ａ）に示すような論理関係を維持しつつ、実体配列の要素自体を変化させることなく、要素の更新をなすことが可能となる。

【００５９】

[値変換配列の生成]

次に、上記値変換配列の作成手順の一例につき、図２５のフローチャートを用いて説明を加える。図２５の例においては、図２３（ｂ）に示すものと同様に、ある数以上の値に所定の数を加えるための値変換配列を生成している。なお、この処理も、図１に示すＣＰＵ１２により実行される。

まず、必要な場合に、初期的な値変換配列が生成される（ステップ２５００）。このステップの処理は、値変換配列が生成されていなかった場合にのみ、実行される。図２３（ｂ）に示す実体配列２３０１に関しては、図２６（ａ）に示すような、初期的な値変換配列が得られる。初期的な値変換配列２６０１においては、終了位置配列中の要素として、実体配列中の要素の最大値が割り当てられ、オフセット配列中の要素として「０」が割り当てられる。

【００６０】

次いで、あるオフセットを与える要素（格納値）の範囲に基づき、生成すべき

配列の行数が決定される（ステップ２５０１）。たとえば、図２３（ａ）に示すように、「１」以上の値をもつ要素に「１」を加えるという条件が与えられた場合には、終了位置配列およびオフセット配列の行数は「２」となる。これは、「１」より小さな値を持つ要素に関する行（或いは構造体配列）、および、「１」以上の値をもつ要素に関する行（或いは構造体配列）が生成されるべきだからである。

その後に、昇順で、各範囲の最大値が終了位置配列の対応する行に配置されるとともに、そのオフセット値が、オフセット配列の対応する行に配置される（ステップ２５０２、２５０３）。図２３（ａ）に示すような実体配列において、「１」以上の値を持つ要素に「１」を加えるべき場合には、終了位置配列において、第０行（１行目）に、１より小さい値の最大値である「０」が配置され、オフセット配列の第０行（１行目）に、オフセット値「０」が配置される。すなわち、第０行の構造体配列に数値が与えられる。その一方、終了位置配列において、第１行（２行目）には、実体配列中の要素の最大値「１」が与えられる。これは、値変換配列の初期値を用いれば良い。また、オフセット配列の第１行（２行目）に、オフセット値「１」が配置される（図２６（ａ）の符号２６０２参照）。

上記ステップ２５０２、２５０３の処理を条件にしたがった全ての範囲に施すことにより（ステップ２５０４、２５０５参照）、値の変換条件にしたがった値変換配列を得ることが可能となる。

【００６１】

なお、上記実施の形態では、１つの条件により、２つの変換配列の領域が形成されたが、これに限定されるものではなく、複数の条件（たとえば、０≦値≦１である場合には、要素に「１」を加え、１≦値≦４である場合には、要素に「－１」を加え、その他の要素に関してはオフセットを与えない）に基づいて、値変換配列を作成できることは明らかである（図２６（ｂ）の符号２６１１、２６１２参照）。

上記添え字変換配列および値変換配列は、組み合わせて利用することができる。たとえば、図２７においては、「０」ないし「９９」の要素が繰り返し現れる実体配列（符号２７００参照）の上流側（入力側）に、「０」から「５００，０

００」までの添え字にはオフセット「０」を与え、「５００，０００」から「９９９，９９９」までの添え字にはオフセット「１」を与える添え字変換配列（符号２７０１参照）を設け、実体配列の下流側（出力側）に、値が４９以下であればオフセット「－１」を与え、値が５０以上であればオフセット「１」を与える値変換配列（符号２７０２）を設けている。これにより、添え字が、添え字変換配列、実体配列および値変換配列を経て、出力値を得ることができる。

【００６２】

上記添え字変換配列を用いて、ある情報ブロック中の実体配列に与えられる添え字を変換し、変換済の添え字により実体配列中の要素（項目値）を取り出し、さらに、上記値変換配列を用いて、取り出された要素（項目値）を変換することにより、当該情報ブロックの入力（添え字）と出力（変換済の値）とからなる見かけの（論理的な）配列を考えることが可能となる。

【００６３】

[具体的な表形式データに対するデータの削除]

次に、上述した手法を用いて、実際の表形式データの所望のデータを挿入し、削除し或いは更新する具体的な手順につき説明を加える。第５の実施の形態では、会員姓および会員名という項目からなる人名テーブルに、所望の会員姓名が挿入、削除され、或いは、その姓或いは名が更新される。ここでは、挿入に関して第２の実施の形態にかかる手法を用いている。すなわち、添え字変換配列および値変換配列が、それぞれ、終了位置配列およびオフセット配列から構成されるようになっている。なお、以下に述べる添え字変換配列の生成処理、および、値変換配列の生成処理には、新たに変換配列を生成することと、もともと存在した変換配列を更新することとが含まれる。したがって、以下において、「生成」は、生成や更新を意味し、場合によっては、これを「生成（ないし更新）」と称する。

【００６４】

図２８（ａ）は、人名テーブルの例を示す図である。このような人名テーブル２８００のために、本実施の形態においては、図２８（ｂ）に示すレコード番号リスト２８１０、図２８（ｃ）に示す「会員姓」という項目に関する情報ブロッ

ク２８２０、および、図２８（ｄ）に示す「会員名」という項目に関する情報ブロック２８３０が生成される。情報ブロック２８２０の値リストへのポインタ配列２８２１において、初期的な添え字変換配列２８２３および初期的な変換配列２８２４は予め作成されている。また、実体配列２８２５が、変換配列を作成する前の（もとの）値リストへのポインタ配列に対応する。値リスト２８２２においても、初期的な添え字変換配列２８２６が予め作成されている。なお、実体配列２８２７が、もともとの項目値を格納した値リストに対応している。

情報ブロック２８３０の値リストへのポインタ配列２８３１においても、初期的な添え字変換配列および値変換配列が予め作成されており、値リスト２８３２においても、初期的な添え字変換配列が予め作成されている。

【００６５】

このような構造を有する情報ブロックからなる表形式データにおいて、図２９に示すように、３つあるレコードのうち（図２９（ａ）参照）、第１行（２行目）のレコードの削除する（図２９（ｂ）参照）場合を考える。この場合には、以下に述べるような各情報ブロックに関して、［添え字変換配列による要素の削除］にて述べた処理が実行される。

まず、「会員姓」の情報ブロックに関して考察を加える。

図２９（ａ）のような「会員姓」を得るために、初期的には、値リストへのポインタ配列において、図３０の符号３０００に示すような添え字の入出力関係が成立している。すなわち、添え字である「レコード番号」の入力に対して、実体配列である「値リストへのポインタ配列」のポインタ値が、符号３０００に示すように出力される。

【００６６】

さて、このような「値リストへのポインタ配列」中の添え字変換配列に対して、図１に示すＣＰＵ１２により、図１７に示すものと略同様の処理が施される。より具体的には、まず、実体配列（ポインタ配列）３００１において削除すべき要素の位置等が特定され（図１７のステップ１７０１）、削除位置に関する変換配列の領域が確保される（ステップ１７０２）。次いで、削除位置の前に関して、終了位置配列の要素として、「削除位置－１」すなわち「０」が配置される（

ステップ１７０３）。なお、この例では、開始位置配列が設けられていないため、ステップ１７０４およびステップ１７０５が省略される。

その後に、削除位置より後に関して、終了位置配列の各要素がデクリメントされる（ステップ１７０６）とともに、オフセット配列の各要素がインクリメントされる（ステップ１７０７）。このような処理により、図３０（ｂ）に示すような、添え字変換配列を得ることができる。図３０（ｂ）において、添え字であるレコード番号に対して、実体配列である「値リストへのポインタ配列」のポインタ値が、符号３０１０に示すように出力されることが理解できるであろう。

【００６７】

また、「会員名」に関する情報ブロック（図２８の符号２８３０参照）に関しても、「会員姓」に関する情報ブロックと同様に、ＣＰＵ１２により、図１７に示すものと略同様の処理が施される。これにより、図３１（ａ）に示す「会員名」の情報ブロック中の添え字変換配列３１０１が、図３１（ｂ）に示すような添え字変換配列３１１１になる。したがって、添え字である「レコード番号」の入力と、実体配列である「値リストへのポインタ」のポインタ値との関係は、図３１（ａ）の符号３１００に示すものから、図３１（ｂ）の符号３１１０に示すものに変化する。

【００６８】

このように添え字変換配列の生成処理を施した後に、レコード番号からポインタ配列を経ることにより、値リストの実体配列中において所望の要素（項目値）を特定することができる。図３２に示すように、ＣＰＵ１２は、新たなレコード番号（添え字）から、添え字変換配列を経た新たな添え字により、実体配列中の要素（ポインタ値）を特定する。たとえば、レコード番号「０」に関するオフセット値は「０」であるため、実体配列中の第０行（１行目）の要素（ポインタ値）が取り出され、このポインタ値「１」によって、値リスト中の実体配列における第１行（２行目）の項目値が特定される（図３２において①参照）。これに対して、レコード番号「１」に関するオフセット値は「１」であるため、「１＋１＝２」という変換済の添え字が得られ、これにより、実体配列中の第２行（３行目）の要素（ポインタ値）が取り出される。次いで、このポインタ値「１」によ

って、値リスト中の実体配列における第１行（２行目）の項目値が特定される（図３２において②参照）。

【００６９】

「会員名」の情報ブロックに関しても、同様の経路（③、④参照）により、値リスト中の実体配列の項目値が特定されることは明らかであろう。このようにして、新たなレコード番号に対応する人名テーブルの表（ビュー）３２００を得ることが可能となる。ビュー３２００において、①ないし④は、上記処理において特定された項目値をそれぞれ示す。

【００７０】

[具体的な表形式データに対するデータの挿入]

次に、データを挿入する例につき説明を加える。たとえば、２つあるレコード（図３３（ａ）参照）のうち、第０行（１行目）と第１行（２行目）との間に、あるレコードを挿入する（図３３（ｂ）参照）を考える。この場合には、以下に述べるような各情報ブロックに関して、「添え字変換配列による要素の挿入が実行されるとともに、必要な情報ブロックに関して「値変換配列の生成」が実行される。

図３４は、レコード（データ）の挿入前の各情報ブロックの状態を示す図である。まず、「会員姓」の情報ブロックにおける添え字変換配列の更新、および、「会員名」の情報ブロックにおける添え字変換配列の更新につき考察を加える。

【００７１】

「会員姓」に関しては、値リストの実体配列中に、「佐藤」という要素（項目値）を加える必要がある。そこで、ＣＰＵ１２は、値リスト中の添え字変換配列による挿入処理（図１３参照）を実行する。より具体的には、要素（項目値）の挿入位置が決定される（図１３のステップ１３００参照）。この例では、「佐藤」という要素（項目値）を挿入する必要があるが、値リストの実体配列においては、要素が昇順（この場合には五十音順）で配置されているため、「佐藤」という要素は、「鈴木」という要素の前に配置する必要があり、したがって、その挿入位置は第０行（１行目）と判断される。

そこで、図１３のステップ１３０１にしたがって、各変換配列の領域が確保さ

れる。この例では、挿入位置が第０行（１行目）であるため、挿入位置より前には要素が存在せず、挿入位置および挿入位置より後の要素に対して、ステップ１３０３ないしステップ１３０７の処理が施される。図３５（ｂ）に示すように、挿入位置（符号３５１０参照）に関して、終了位置配列に、挿入位置を示す要素「０」が配置され（符号３５１１参照）、かつ、実体配列の末尾を示す位置「１」が、オフセット配列の対応する要素として配置される（符号３５１２参照）。また、挿入位置より後に関して、終了位置配列に、もとの値に「＋１」を加算した値が配置され（符号３５１３参照）、オフセット配列の対応する値として、もとのオフセット値「０」に「－１」を加算した値が配置される（符号３５１４参照）。また、実体配列の末尾に、新たな要素（項目値）「佐藤」が配置される（符号３５２０参照）。実体配列においては挿入すべき新たな要素が末尾に付加された形態となっているが、値リスト中の添え字変換配列により、見かけの（つまり論理的な）配列は、符号３５２１にて示すように、昇順（五十音順）となっている。

【００７２】

同様に、「会員名」に関しても、値リストの実体配列中に、「エー作」という要素（項目値）を加える必要がある。そこで、ＣＰＵ１２は、「会員名」の情報ブロックにおいて、の値リスト中の添え字変換配列による挿入処理（図１３）を実行する。より具体的には、要素（項目値）の挿入位置が、第１行（２行目）であると決定され（図３６（ｂ）の符号３６１０参照）、各変換配列の領域が確保される。挿入位置より前の要素に対しては、その終了配列中の要素として、挿入位置（この場合「１」）から「１」を減じた（「－１」を加算した）値が割り当てられ（符号３６１１参照）、その一方、対応するオフセット配列中の要素は保持される（符号３６１２参照）。

また、挿入位置（符号３６１０参照）に関して、終了位置配列に、挿入位置を示す要素「１」が配置され（符号３５１３参照）、かつ、実体配列の末尾を示す位置「２」を示すためのオフセット値「１」が、オフセット配列中の要素として与えられる（符号３６１４参照）。

【００７３】

その一方、挿入位置より後に関して、終了位置配列に、もとの値に「＋１」を加算した値が配置され（符号３６１５参照）、オフセット配列の対応する値として、もとのオフセット値「０」に「－１」を加算した値が配置される（符号３５１６参照）。さらに、実体配列の末尾に、新たな要素（項目値）「エー作」が配置される（符号３６２０参照）。なお、実体配列においては挿入すべき新たな要素が末尾に付加された形態となっているが、値リスト中の添え字変換配列により、見かけの（つまり論理的な）配列は、符号３６２１にて示すように、昇順（五十音順）となっていることが理解できる。

このようにして各情報ブロックの値リストに関する値変換配列の生成（ないし更新）が実現された後、これに付随した、ポインタ配列の側の添え字変換配列および値変換配列の生成（ないし更新）が実行される。

【００７４】

まず、値リストへのポインタ配列において、実体配列の要素を変換するための値変換配列が生成（ないし更新）される必要がある。たとえば、図３７に示す「会員姓」の情報ブロックにおける値リストへのポインタ配列に関して、関連する値リスト（すなわち「会員姓」の値リスト）に、先の処理により、新たな姓「佐藤」が見かけの（つまり論理的な）配列の先頭に挿入されている（図３５の符号３５２１参照）。そこで、値リストへのポインタ配列のポインタ値（すなわち実体配列３７０１における要素）のうち、０以上の値を有するものがすべてインクリメントされなければならない。そこで、ＣＰＵ１２は、図２５に示すフローチャートにしたがって、上記値リストへのポインタ配列における値変換配列を生成（ないし更新）する。

【００７５】

図３７（ａ）に示す配列では、値のとり得る最大値が「０」であり、かつ「０」以上の値をインクリメントする必要があるため、結果として、オフセット配列中の値「０」に「＋１」が加算される（図２５のステップ２５０３、図３７（ｂ）の符号３７１１参照）。初期的には、実際に実体配列中に収容された要素（ポインタ値）と、値変換配列を経て出力される要素（ポインタ値）とが、図３７（ａ）の符号３７０２に示すように同一であったのに対して、値変換配列の生成（

ないし更新）の後には、値変換配列を経て出力される要素（ポインタ値）が、図３７（ｂ）の符号３７１２に示すように、もとの値をインクリメントした形態となる。

また、新たな姓「佐藤」がレコード番号「１」（すなわち第１行或いは２行目）に挿入されることに伴って、値リストへのポインタ配列における添え字変換配列の生成（ないし更新）処理を実行するとともに、値リスト中の見かけの（つまり論理的な）配列の第０行（１行目）にも姓「佐藤」が挿入されたため、値リストへのポインタ配列中の値を、上記値リストの変更に伴って生成（ないし更新）する処理を実行する必要がある。

つまり、値リストへのポインタ配列では、添え字であるレコード番号から整数化されたポインタ値を取り出すものであり、したがって、以下の添え字変換配列の生成（ないし更新）により、添え字変換配列を生成して、レコード番号「１」に対応する姓「佐藤」を示すことができるようにするとともに、ポインタ配列からの出力（値）を変換して、「佐藤」および「鈴木」という要素を昇順で並べた見かけの（つまり理論的な）値リストを適切に示すことができるようにしている。

【００７６】

まず、ＣＰＵ１２は、上記ポインタ配列の添え字変換配列に関して、添え字であるレコード番号が挿入されたことに対応して、図１３に示す処理を実行する。

より具体的には、新たな姓「佐藤」（および新たな名「エー作」）を挿入すべきレコード番号が「１」であるため、挿入位置が第１行（２行目）であると判断され（図１３のステップ１３０１参照）、挿入位置およびその前後に関する変換配列の領域が確保される（ステップ１３０２）。以下、ステップ１３０３ないしステップ１３０８の処理（実際には、開始位置に関する処理が省略される）により、図３８（ａ）に示すように添え字変換配列（符号３８１０参照）が形成される。

【００７７】

これにより、添え字変換配列を経ることにより、値リストのポインタ配列は、添え字であるレコード番号の入力に対して、符号３８１１に示すような出力が選

られることになる。ところで、図３７を参照して説明したように、上記例では、値リストにおける見かけの（つまり理論的な）配列では、昇順（五十音順）にしたがって、「佐藤」という要素（項目値）が「鈴木」という要素（項目値）より上位に位置していることがわかっている。したがって、値変換配列により、値リストのポインタ配列中の実体配列の値を変換して、値リスト中の要素（項目値）を適切に指示できるようにしなければならない。

そこで、ＣＰＵ１２が図２５にしたがった処理を実行する。この処理では、姓「佐藤」の挿入により、実体配列の値が「１」であるものの出力が「０」となるようにオフセット値、すなわち、オフセット値「－１」を与え、それ以外についてはオフセット値「１」を与えるような処理を実行する。これにより、図３８（ｂ）に示すように値変換配列（符号３８２０参照）を得ることができる。

【００７８】

図３８（ｂ）において、新たなレコード番号（添え字）に対して、添え字変換配列、実体配列および値変換配列を経て出力された値は、見かけの（つまり倫理的な）配列（符号３８２１参照）に示すようになり、これにより、適切に値リストの要素（項目値）が特定されることが理解できよう。

「会員姓」の情報ブロックの値リストへのポインタ配列に対して施されたものと同様の処理が、「会員名」の情報ブロックに対しても施される。まず、図３９に示すように、「会員名」の情報ブロックのうち、値リストに関する処理（添え字変換配列の生成（更新）および実体配列への要素（項目値）の挿入：図３６参照）に伴う値リストへのポインタ配列の生成（ないし更新）が実行される。ＣＰＵ１２が図１３に示す処理を事項することにより、まず、図３９（ｂ）に示す値変換配列（符号３９１０参照）を得ることができることは明らかであろう。

【００７９】

次いで、ＣＰＵ１２は、上記ポインタ配列の添え字変換配列に関して、添え字であるレコード番号が挿入されたことに対応して、図１３に示す処理を実行する。より具体的には、新たな会員名「エー作」が挿入すべきレコード番号が「１」であるため、第１行（２行目）を挿入位置として（図１３のステップ１３０１参照）、図１３にしたがった処理が行われる。これにより、図４０（ａ）に示すよ

うなオフセット配列（符号４０１０参照）を得ることができる。

さらに、上記添え字変換配列の生成（ないし更新）および実体配列の末尾への要素の挿入に伴って値変換配列を更新する必要があるため、ＣＰＵ１２は、図２５に示す処理を実行し、これにより、図４０（ｂ）に示す値変換配列（符号４０２０参照）を得ることができる。

【００８０】

図４０（ｂ）において、レコード番号を添え字とする値リストのポインタ配列の出力は、見かけの（論理的な）配列（符号４０２１参照）に示すものとなる。各レコード番号からは図４０（ｂ）の破線で示すような経路で出力が得られていることが理解できよう。

このように、各情報ブロックにおいて添え字変換配列および／または値変換配列の生成ないし更新処理を施した後に、レコード番号からポインタ配列を経ることにより、値リストの実体配列中において所望の要素（項目値）を特定することができる。図４０に示すように、ＣＰＵ１２は、値リストへのポインタ配列において、新たなレコード番号（添え字）から、添え字変換配列、実体配列および値変換配列を経て、値リストの実体配列中の要素（項目値）を特定するための出力を得る。たとえば、「会員姓」の情報ブロックに関して、レコード番号「０」に関するオフセット値は「０」であるため、実体配列中の第０行（１行目）の要素（ポインタ値）が取り出され、さらに、この要素（ポインタ値）の対応するオフセット値が「１」であるため、値リストへのポインタ配列からの出力は「１」となる（図４０において①参照）。これに対して、レコード番号「１」に関して、添え字変換配列のオフセット値は「１」であるため、「１＋１＝２」という変換済の添え字が得られ、これにより、実体配列中の第２行（３行目）の要素（ポインタ値）が取り出される。さらに、この要素（ポインタ値）に対応するオフセット値が「－１」であるため、値リストへのポインタ配列からの出力は「０」となる（図４０において②参照）。同様にしてレコード番号「３」についても、出力「１」を得ることができる。このような出力は、値リストに与えられ、値リスト中の添え字変換配列を経たものにより、実体配列中の要素（項目値）が特定される。「会員名」の情報ブロックに関しても、同様の経路を経て、レコード番号か

ら値リストの実体配列中の要素（項目値）が特定される（図４１の④ないし⑥参照）。

【００８１】

このようにして、新たなレコード番号に対応する人名テーブルの表（ビュー）４１００を得ることが可能となる。ビュー４１００において、①ないし⑥は、上記処理において特定された項目値をそれぞれ示す。

本実施の形態によれば、実体配列への入力を添え字として定義し、当該添え字を変換し、および／または、実体配列からの出力である値を変換することにより、実体配列の要素自体を移動することなく、実体配列への要素の挿入および削除を実現している。つまり、実体配列への入力および出力に所定の関数を施すことにより、仮想的に（或いは論理的に）実体配列に対する挿入／削除／更新が行われたとみなせるように構成している。したがって、実体配列中の要素の移動を必要とすることなく、データの挿入等を実現することができる。特に、要素数が膨大な表形式データにおいて、要素の移動を伴わないことにより、処理の著しい高速化に寄与することが可能となる。

【００８２】

[表形式データに対するデータの更新]

上述したように、本発明においては実体配列の変更（すなわち、データ自体の変更や移動）を伴うことなく、表形式データに対するデータの挿入および削除を実現することができる。表形式データに対するデータの更新は、あるレコード番号のデータの削除（レコードの削除）および同じレコード番号へのデータの挿入（レコードの挿入）を順次実行することと等価である。たとえば、図２８に示す表形式データ（人名テーブル）において、第１行（２行目）のレコードを更新する行為（たとえば、「会員姓」の「佐藤」を「田中」に更新する処理、および／または、「会員名」の「エー作」を「ビー作」に更新する処理）を考える。この場合に、まず、図２９ないし図３２を参照して説明した手法を用いて、第１行（２行目）のレコードを削除する処理を実行し、その後に、図３３ないし図４２を参照して説明した手法を用いて、第０行（１行目）と第１行（２行目）との間に、新たなレコードを挿入する処理を実行すれば良い。

【００８３】

[情報ブロックの形態にて表わされた表形式データの挿入等に関する考察]

さて、上記情報ブロックの形態にて表わされた表形式データにおいて、データを追加する場合につき、再度考察を加える。

前述したように、上述した例では、図４６（ａ）に示すように、ＣＰＵ１２により、値リストへの項目値の挿入に伴う添え字変換配列の処理、すなわち、五十音順にソートされた状態にて項目値が収容された状態の「見かけの（論理的な）配列」を得るために添え字変換配列を生成（ないし更新）する処理が実行される（ステップ４６０１）。その後に、上記処理において、見かけの（論理的な）実体配列（値リスト）の所定の位置に、要素（項目値）が挿入されているため、これに伴って、値リストへのポインタ配列に関して、所定の位置に対応する値以上の値を有するものがインクリメントされている（ステップ４６０２）。次いで、レコード番号とポインタ配列中のポインタ値に着目して、ＣＰＵ１２は、レコード番号の挿入に伴って、値リストへのポインタ配列の添え字変換配列の処理を実行している（ステップ４６０３）。このような処理の後、最後に、新たに挿入されたポインタ配列の実体配列における項目値（前記例では、もとの実体配列中の最大値よりも一つ大きな値）に対する値変換配列を生成している（ステップ４６０４）。

【００８４】

この処理を見ると、ステップ４６０１とステップ４６０３とは独立して実行でき（場合によっては並列的に実行することもでき）、その一方、ステップ４６０２とステップ４６０４とは一括して実行することができる。したがって、挿入処理は、図４６（ｂ）に示すように、ポインタ配列の添え字変換配列に関する処理（ステップ４６１１：図４６（ａ）のステップ４６０３に対応する）、値リストの添え字変換配列に関する処理（ステップ４６１２：ステップ４６０１に対応する）、および、ポインタ配列の値変換に関する処理（ステップ４６１３：ステップ４６０２および４６０４に対応する）からなると考えることができる。

さて、ここで、ｍ個のレコードに対応してｍ個の要素（ポインタ値）からなる値リストへのポインタ配列と、ｎ個の要素（項目値）からなる値リストとを考え

る。ここで、第ｊ行のレコードに、要素を挿入すると仮定する。

図４７（ａ）に示すように、第ｊ行の要素の挿入に対して、ステップ４６１１の処理により、値リストへのポインタ配列に関して、終了位置配列およびオフセット配列からなる添え字変換配列（符号４７０１参照）、および、その末尾に新たなポインタ値が格納された実体配列（符号４７０２参照）が得られることが理解できるであろう。

【００８５】

その一方、所定の順序（たとえば五十音順）にて要素（項目値）が配置された値リストに関して、この要素を走査することにより、第ｉ行に要素を挿入すべきであると判断された場合にも、ステップ４６１２の処理により、終了位置配列およびオフセット配列からなる添え字変換配列（符号４７０３参照）、および、その末尾に挿入すべき要素（項目値）が配置された実体配列（符号４７０４参照）が得られることが理解できるであろう。ここで、値リストへのポインタ配列において、添え字変換配列は、添え字（レコード番号）「ｐ」が、０≦ｐ≦（ｊ－１）である場合には、オフセット値「０」を加え、ｊ≦ｐ≦ｊ（すなわちｐ＝ｊ）の場合には、オフセット値「ｍ－ｊ」を加え、或いは、（ｊ＋１）≦ｐ≦ｍの場合には、オフセット値「－１」を加えることを意味している。また、値リストにおいて、添え字変換配列は、入力した添え字（ポインタ配列の出力）「ｑ」が、０≦ｑ≦（ｉ－１）である場合には、オフセット値「０」を加え、ｉ≦ｑ≦ｉ（つまりｑ＝ｉ）の場合には、オフセット値「ｎ－ｉ」を加え、或いは、（ｉ＋１）≦ｑ≦ｎの場合には、オフセット値「－１」を加えることを意味している。

【００８６】

次に、値リストへのポインタ配列における値変換配列に考察を加えると、値リストの見かけ（論理上）の挿入位置「ｉ」より小さな値についてはオフセットを加える必要が無い（すなわち、ポインタ値の意味合いに変化が無い）と考えることができる。その一方、上記挿入位置「ｉ」以上の値については、見かけ（論理上）の値リストに対してポイントすべき位置が、一つずつ下方（すなわち値の大きな方）にシフトしていることになるため、その値にはオフセット値「１」を加えるべきであることが理解できる。

さらに、値リストの実体配列の末尾に追加されたポイント値（すなわち、レコード番号の挿入処理に伴って付加された、実体配列中の位置「ｍ」におけるポインタ値）の値は、「ｎ」であると考えられる。これは、挿入処理前のポインタ配列の実体配列において、「０」ないし「ｎ－１」の何れかの値を取り得るため、挿入処理によりポイント値「ｎ」が与えられるからである。このため、このポイント値「ｎ」が値変換により、値リストの見かけ（論理上）の挿入位置「ｉ」をポイントするためには、そのオフセット値を「ｉ－ｎ」に設定すればよいことがわかる。無論、実体配列中の位置「ｍ」におけるポイント値が他の値を割り当てても、値変換により、その値が「ｉ」に変換されるようなオフセット値を与えればよい。このようにして、図４７の符号４７０５に示すような値変換配列を得ることが可能となる。すなわち、ポインタ配列の値変化リストに関して、実体配列中の要素（ポインタ値）「ｒ」が、０≦ｒ≦(ｉ－１)である場合には、オフセット値０を加え、ｉ≦ｒ≦(ｎ－１)の場合には、オフセット値「＋１」を加え、或いは、ｎ≦ｒ≦ｎ（つまりｒ＝ｎ）の場合には、オフセット値「ｉ－ｎ」を加えればよいことになる。

【００８７】

削除に関しても、挿入の場合と同様に、ｍ個のレコードに対応してｍ個の要素（ポインタ値）からなる値リストへのポインタ配列と、ｎ個の要素（項目値）からなる値リストを考え、その第j行のレコードを削除すると仮定する。この場合には、図４８に示すように、ポインタ配列中の添え字変換配列（符号４８０１）参照を得ればよい。図４８から理解できるように、添え字（レコード番号）「ｐ」が、０≦ｐ≦(ｊ－１)の場合には、オフセット値０を加え、ｊ≦ｐ≦(ｍ－２)の場合には、オフセット値「＋１」を加えればよい。これにより、変換後の添え字は、もとの（変換前の）添え字「ｊ」に対応するポインタ値を、ポインタすることがなくなり、これにより当該ポインタ値およびこれにより特定される値リストが、見かけ上（論理的に）削除される。

【００８８】

[他の手法による表形式データに対するデータの削除]

次に、本発明において、表形式データに対するデータの削除の他の例につき説

明を加える。この手法は、データの挿入がなされない場合に有効である。再度、図２８のような構造を有する情報ブロックからなる表形式データにおいて、図２９に示すように第１行（２行目）のレコードを削除する場合を考える。

【００８９】

本例では、「会員姓」および「会員名」の情報ブロックに処理を施すことなく、レコード番号リストに処理を施すことにより、レコードの削除を実現している。より詳細には、図４２に示すように、レコード番号リストの添え字変換配列に関して、削除の処理を実行し、もとのレコード番号「１」を削除した形態の論理的な出力をなすようにしている。図４２（ａ）に示すように、レコード番号リストにおける添え字入力とレコード番号出力との関係は、符号４２０１に示すようなものとなっている。そこで、ＣＰＵ１２は、第１行（２行目）を削除するために、図１７に示す処理を実行する。より詳細には、削除位置（この場合には第１行）が特定され（ステップ１７０１）、次いで、削除位置に関する変換配列の領域が生成され（ステップ１７０２）、変換配列中の要素につき、所定の処理が実行される（ステップ１７０３ないし１７０７参照、ただし、この例では、開始配列に関する処理は実行されていない。）。これにより、図４２（ｂ）に示すような添え字変換配列（符号４２１０参照）が得られる。この添え字変換配列を経ることにより、入力と出力の関係は符号４２１１に示すようなものとなり、論理的にはレコード番号「１」が削除され、それ以下のレコード番号が一つずつシフトした状態が実現されていることが理解できるであろう。

本例によれば、レコード番号自体の添え字変換配列を用いることにより、仮想的（論理的）にレコード番号の削除およびシフトが実現され、したがって、極めて簡素な手法によりレコードの削除を実現することが可能となる。

【００９０】

［本発明のトランザクション処理等への応用］

次に、上記手法を用いてトランザクションを適切になすことにつき説明を加える。上述したように、本発明においては実体配列の要素の順序を変更したり、或いは、要素を移動することなく、要素の挿入、削除並びに更新を実現している。そこで、これを用いれば、トランザクションを適切に実現することができる。図

４３は、トランザクション処理を示すフローチャートである。この処理は、図１のＣＰＵ１２により実行される。また、表形式データを複数の情報ブロックの集合体の形態とするための処理（たとえば、図６に示す処理）は予め実行されているものとする。

【００９１】

ＣＰＵ１２は、まず、トランザクション処理に関わる全ての情報ブロックに関して初期的な変換配列を付与する（ステップ４３０１）。たとえば、レコード番号リストにおいては、レコード番号の配列（実体配列）に対する初期的な添え字変換配列を生成し、或いは、ある情報ブロックの値リストへのポインタ配列においては、ポインタ配列の実体配列に対する初期的な添え字変換配列および値変換配列を生成するとともに、当該情報ブロックの値リストにおいては、実体配列（実際の値リスト）に対する初期的な添え字変換配列を生成する。

次いで、ＣＰＵ１２は、所定のトランザクション処理を実行する（ステップ４３０２）。ここにおいて、レコードの挿入、削除或いは更新の何れかが必要である場合には、前述したように、添え字変換配列や値変換配列を更新し、かつ、必要な場合には実体配列の末尾に要素を挿入（或いは別の領域に差分配列を生成）する。たとえば、オペレータ等によりロールバックの指示があった場合（ステップ４３０３でイエス(Yes)には、ＣＰＵ１２は、その時存在した全ての変換配列を廃棄する（ステップ４３０４）。

【００９２】

その一方、ロールバックの指示がなく（ステップ４３０３でノー(No)）、かつ、トランザクション処理が終了した場合には、ＣＰＵ１２は処理をコミットする（ステップ４３０４）。コミットの際に、ＣＰＵ１２は各変換配列を破棄しても良いし、或いは、各変換配列の状態を、ステップ４３０１にて生成した初期的な変換配列のものに戻し、他のプロセスがこの状態の変換配列を経るように構成しても良い。また、処理終了時の変換配列の状態のまま、他のプロセスがこれを利用しても良い。さらに、変換配列を維持して、必要に応じてこれらを利用できるようにしても良い。

【００９３】

次に、同じ表形式データを用いた複数のプロセスが並列的に実行される場合を考える。たとえば、あるプロセスがデータの更新を伴っており（たとえば、口座から１万円を引き出す処理）、その一方、他のプロセスが表形式データを参照している（たとえば、各口座の預金残高の総領を求める処理）場合を考える。

上記データの更新を伴うプロセス（「更新プロセス」と称する）においては、コミットが生じるまで、１万円が仮に引き出された状態となる。本発明においては、更新プロセスが、添え字変換配列や値変換配列を用いてレコードの削除および挿入を行っており（図４４の実線の矢印参照）、その一方、他のプロセス（「参照プロセス」と称する）は、上記添え字変換配列や値変換配列を参照することなしに、直接実体配列を参照する（図４４の破線の矢印参照）ことにより、複数の処理が並列的に実行されても、それぞれが干渉することなく適切に処理をなすことができる。

【００９４】

次に、レコードのロックの手法につき簡単に説明を加える。この場合には、各レコード番号に対応つけて、ロックを管理する情報ブロックを設ければ良い。図４５は、レコードのロックを説明するための図である。図４５に示すように、表形式データにおいて、ロック管理の情報ブロック（符号４５０１）が設けられ、当該情報ブロック中に、レコード番号数の要素を配置できるポインタ配列（符号４５０２）を設けている。このポインタ配列には、要素（格納値）として、読み書き不可能を示す「ＮＯＮ」、読み出し専用（更新や削除の不可）を示す「Ｒ」、読み書き可能を示す「ＲＷ」などが格納されている。無論、このような文字を用いる必要なく、それぞれのステータスを示す数値（たとえば、読み書き不可能であれば「０」、読み出し専用であれば「１」など）を割り当てれば良いことは言うまでもない。このように、レコード番号に対応したポインタ配列を参照することにより、プロセスは当該レコード番号に対応する種々のデータの読み書き可能性を知ることができるとともに、プロセスが、処理の際に、所定のレコード番号に対応するポインタ配列を更新することにより、排他処理を実現することが可能となる。

【００９５】

このように、各レコードに対応してロックの属性（種別）を含む項目値を配置した配列を設けることにより、まず、大量のレコードに対して、高速にロックをなすことが可能となる。たとえば、ある項目に関して特定の項目値を有するレコード（たとえば、性別という項目で「女性」という項目値を有するレコード）を、高速にロックすることができる。すなわち、従来では、ＯＳコールにてロックしたいレコードを一つ一つＯＳに登録すべきところ、本発明によれば、情報ブロックの配列中の項目値を変更することで、ＯＳコールを介することなく、そのロックをなすことが可能となる。

また、上記構成によれば、表形式データをジョインさせた場合にも、そのロックを伝播することが可能となる。たとえば、特願平１１－１５１１５６号にしたがって、各々が複数の情報ブロックの集合体である複数の表形式データのうち、共通の意味合いを有する情報ブロックの項目値を共通化することにより、ジョインを実現する場合を考える。まず、各表形式データに関して、ロックに関する情報を格納する配列を含む情報ブロックが生成される。ここに、ジョインの処理をなす前の各表形式データの情報ブロック中の配列を原テーブルと称する。次いで、ジョインの処理がなされるが、この場合、原テーブルのポインタ配列は、基本的に保存されるため、ジョインにより生成されたテーブルから、原テーブルのロック情報を格納する配列を含む情報ブロックを参照ないし書き換えすることが可能である。

【００９６】

したがって、ジョインテーブルを作成しこれを表示しながら、ユーザからのデータ変更以来を受け付けることが可能となる。より具体的に、以下にその手順を説明する。

（１）まず、ジョインテーブルを作成し、これが表示される。

（２）次いで、ユーザがあるジョインテーブル中のレコードを変更することを指示すると、プログラムは、ユーザが指示したレコードにロックがかかっていないことを確認する。このときに、ジョインにより生成されたテーブルには、ロック情報を管理する情報ブロックが存在しないため、実際には、原テーブルにおける該当レコードに関するロック情報を調べる。

（３）次いで、プログラムは、ジョインテーブル中のそのレコードをロックする。実際には、原テーブルの対応するレコードをロックすることになる。

（４）その後に、プログラムは、ユーザの指示にしたがってデータの書き換えを実行する。

（５）最後に、プログラムは、変更をコミットし、ロックを解除する。

このように、上記構成に従えば、ロックをジョインテーブルにも伝播できるため、ジョインテーブルを書き換え可能なものとすることが可能となる。

【００９７】

本発明は、以上の実施の形態に限定されることなく、特許請求の範囲に記載された発明の範囲内で、種々の変更が可能であり、それらも本発明の範囲内に包含されるものであることは言うまでもない。

たとえば、本発明は、前記第５ないし第７の実施の形態のように、複数の情報ブロックの組により構築された表形式データに対するデータの挿入、削除および更新に限定されるものではない。図４９に示すように、レコード番号（符号４９０１参照）と、姓（４文字）、名（４文字）、住所（２０文字）等の文字列からなる構造体（符号４９０２参照）とが対応付けられた形態の表形式データに関するデータの挿入、削除および更新をなすこともできる。

たとえば、挿入の場合には、氏名、住所等ならなる構造体配列の組を実体配列と考え、レコード番号を添え字と考えて、図１３に示すものと略同等の処理を実行すれば良い。また、削除の場合にも、同様に、構造体配列の組を実体配列と考え、レコード番号を添え字と考えて、図１７に示すものと略同等の処理を実行すれば良い。

【００９８】

また、本発明にかかる値変換配列を単独で利用することも可能である。たとえば、Ｄ／Ａ変換器（図示せず）から、順次１億サンプルの測定データが与えられる場合を考える。通常、１億サンプルのデータを一つのメモリにて収容することは容易でなく（たとえば、２バイト／１サンプルであっても、２億バイト（１００Ｍバイト以上）のデータ容量を要する）、これを１０個のメモリに分割して記憶することを考える。図５０において、先頭（第０番）のメモリブロック５００

０から最後尾（第９番）のメモリブロック５００９に、それぞれ１０００万個の測定データ（サンプルデータ）が収容されている。

【００９９】

このような場合には、各測定データの累算値を求め、これを記憶すべき場合を考える。本発明に関して、これは以下の手順により実行することができる。

各メモリブロックに関して、その先頭から末尾までの測定データのデータ値を加算する。これは、複数のＣＰＵ（この場合には１０個が望ましい）が並列的に、自己が演算をなすべきメモリブロックに対して、累算処理を実行すれば良い。これにより、図５０の符号５０１０、５０１１、…、５０１９に示すような各ブロックごとの累算値を算出することができる。

【０１００】

次いで、メモリブロックの上位に隣接するメモリブロック中の最終アドレスに対応する論理上の累算値が取り出されて、これが当該メモリブロックの値変換配列におけるオフセット配列中のオフセット値となる。この例では、一つのメモリブロックにおいて、同一のオフセット値が用いられるため、終了位置配列など他の配列を作成する必要はない。たとえば、メモリブロック５０２１に関して、そのオフセット値は、Ｚ（９，９９９，９９９）（ここで、ｚは、論理上の累算値）となり、たとえば、当該メモリブロック５０２１の先頭アドレスに対応する累算値Ｙ（１０，０００，０００）に、オフセット値Ｚ（９，９９９，９９９）を加算することにより、論理上の累算値Ｚ（１０，０００，０００）を求めることができる。また、当該メモリブロック５０２１の末尾アドレスに対応する累算値Ｙ（１９，９９９，９９９）に、オフセット値Ｚ（９，９９９，９９９）を加算することにより、論理上の累算値Ｚ（１９，９９９，９９９）が求められる。この累算値Ｚ（１９，９９９，９９９）は、下位に隣接するメモリブロックにて用いるオフセット値となることが理解できるであろう。

【０１０１】

ここで、累算に要する時間につき考察を加えると、上述したように１０個のメモリブロックに対して、並列的な累算処理がなされる場合には、各メモリブロックの累算値を求めるために１０００万ステップを要し、また、加算オフセット値

の設定のために、９ステップ程度を要する。これに対して、従来の手法で累算値を求めるためには、１億ステップが必要であることが理解できよう。これは、たとえ、メモリブロックに分割がなされていても、上位に位置するメモリブロックの累算値が算出されて、はじめて当該メモリブロックに関する累算が開始できるため、各メモリブロックに関する並列処理が事実上不可能であることに起因する。このように、本発明の値変換配列を用いることにより、科学技術計算等の高速化も可能となる。

【０１０２】

また、本発明によれば、図５１に示すように、ネスト構造をとることも可能である。図５１において、最も内側に位置するネストにおいては、添え字変換配列（符号５１０１参照）および実体配列（符号５１０２参照）および値変換配列（符号５１０３）が設けられている。その一方、その外側に位置するネストにおいては、上記添え字変換配列、実体配列および値変換配列からなる組を、第２の実体配列（符号５１１２参照）と考え、その入力側に添え字変換配列（符号５１１１参照）を配置するとともに、その出力側に値変換配列（符号５１１３参照）を配置している。同様に、さらにその外側のネストにおいては、内側のネストを第３の実体配列（符号５２１２参照）と考え、その入力側に添え字変換配列（符号５２１１参照）を配置するとともに、その出力側に値変換配列（符号５２１３参照）を配置している。このような構造をとることにより、階層構造の挿入、削除、更新の処理をなすことが可能となる。たとえば、値の挿入を複数回実行して、その後に更新トランザクションとしてコミットする場合に、上記構造は有用である。

【０１０３】

また、前記図４７に示す例では、値リストのポインタ配列において、実体配列の末尾にポインタ値を配置し、そのポインタ値ｆ（ｍ）＝ｎ（ここで、ｎは、今までのポインタ値がとり得る最大値「ｎ－１」よりも１つ大きい。）と設定し、かつ、値リストにおいても、実体配列の末尾に項目値を配置しているが、このような構成に限定されるものではない。すなわち、値リストへのポインタ配列に関して、実体配列に隣接せずに、他の領域（差分配列）にポインタ値を配置しても

よく、或いは、ポインタ値ｆ（ｍ）も、他の値「ｙ」（ただしｙ≧ｎ）と設定してもよい。さらに、値リストに関しても、実体配列に隣接せずに、他の領域（差分配列）に項目値を配置してもよい。図５２は、ポインタ配列に関して、領域「ｘ」にポインタ値（ｆ（ｍ）＝ｙ）を配置し、かつ、値リストに関して、領域「ｚ」に項目値を配置した例を示している。この場合にも、図４７にて説明した場合と略同様に、添字変換配列や値変換配列を求めることが可能である。

【０１０４】

さらに、前記実施の形態において、情報ブロックの形態の表形式データのデータ更新処理は、挿入処理および削除処理、或いは、削除処理および更新処理を順次実行することができることを述べた。このような更新処理に関しても、以下のように、添え字変換配列および値変換配列を求めることができる。

たとえば、ｍ個のレコード番号と、ｎ個の項目値を有する表形式データに関して、レコード番号「ｊ」に関して項目値の更新をすべき場合を考える。ここに、更新すべき項目値が挿入された場合の挿入位置（論理上の挿入位置）が「ｉ」であると考える。

【０１０５】

まず、値リストのポインタ配列において、実体配列の末尾にポインタ値を配置し、そのポインタ値ｆ（ｍ）＝ｎ（ここで、ｎは、今までのポインタ値がとり得る最大値「ｎ－１」よりも１つ大きい。）と設定し、かつ、値リストにおいても、実体配列の末尾に更新に利用される項目値を配置する場合につき説明を加える。この場合には、図５３に示すように、ポインタ配列の添え字「ｐ」に関して、０≦ｐ≦（ｊ－１）である場合には、オフセット値「０」が与えられ、ｐ＝ｊである場合には、オフセット値「ｍ－ｊ」が与えられ、或いは、ｊ＋１≦ｐ≦ｍ－１である場合には、オフセット値「０」が与えられる。また、ポインタ配列の値変換配列、および、値リストの添え字変換配列は、図４７に示すものと同じであることが分かる（削除処理においては、これらに手が加えられないため、挿入処理のものが維持される）。

【０１０６】

或いは、更新処理において、ポインタ配列に関して、領域「ｘ」にポインタ値

（ｆ（ｍ）＝ｙ）を配置し、かつ、値リストに関して、領域「ｚ」に更新される項目値を配置した場合にも、図５２に関して説明したものと同様に、添字変換配列および値変換配列が求められることが理解できよう（図５４参照）。

また、前記実施の形態においては、一般のコンピュータシステム１０内に、所定のプログラムを読み込み、当該プログラムを実行することにより、添え時変換配列や値変換配列の生成、更新等の処理を実現しているが、本発明はこれに限定されるものではなく、パーソナルコンピュータ等のような一般のコンピュータシステムに、データベース処理専用のボードコンピュータを接続し、当該ボードコンピュータが上記処理を実行できるように構成しても良いことは言うまでもない。したがって、本明細書において、手段とは必ずしも物理的手段を意味するものではなく、各手段の機能が、ソフトウェアによって実現される場合も包含する。さらに、一つの手段の機能が、二つ以上の物理的手段により実現されても、若しくは、二つ以上の手段の機能が、一つの物理的手段により実現されてもよい。

【０１０７】

【発明の効果】

本発明によれば、特に、上記線形フィルター法を用いて、データの挿入、削除および更新の処理を高速かつ適切になすことができる表形式データにおけるデータ挿入、削除および更新方法を提供すること可能となる。

本発明は、上記トランザクション処理を適切になすことができる表形式データの挿入、削除および更新方法を提供することが可能となる。

【図面の簡単な説明】

【図１】　図１は、本発明の実施の形態にかかる検索、集計およびサーチ方法を実現できるコンピュータシステムのハードウェア構成を示すブロックダイヤグラムである。

【図２】　図２は、本実施の形態にて用いる情報ブロックを示す図である。

【図３】　図３は、表形式データの例、および、当該表形式データに基づく情報ブロックの例を示す図である。

【図４】　図４は、表形式データの他の例、および、当該表形式データに基づく情報ブロックの他の例を示す図である。

【図５】　図５は、単一項目に関する検索手法を示すフローチャートである。

【図６】　図６は、表形式データに基づき情報ブロックを作成するための処理を説明するフローチャートである。

【図７】　図７は、情報ブロックを作成するための原データの例を示す図である。

【図８】　図８は、配列に要素を挿入する例および配列の要素を更新する例を示す図である。

【図９】　図９は、添え字、配列および配列からの値の関係、および、本発明に係る添え字変換および値変換の概要を説明する図である。

【図１０】　図１０は、要素の挿入に関する入出力の論理的関係と、第１の実施の形態にかかる配列への要素の挿入を説明するための図である。

【図１１】　図１１は、第１の実施の形態にかかる添え字から配列中の要素を特定するための処理を示すフローチャートである。

【図１２】　図１２は、第１の実施の形態において、ある要素の特定の位置への挿入を説明する図である。

【図１３】　図１３は、本実施の形態において要素の挿入のための添え字変換配列を生成するための処理を示すフローチャートである。

【図１４】　図１４は、図１３の処理により生成された添え字変換配列の一例を示す図である。

【図１５】　図１５は、要素の削除に関する入出力の論理的関係と、第１の実施の形態にかかる配列への要素の削除を説明するための図である。

【図１６】　図１６は、第１の実施の形態において、特定の位置の要素の削除を説明する図である。

【図１７】　図１７は、本実施の形態において要素の削除のための添え字変換配列を生成するための処理を示すフローチャートである。

【図１８】　図１８は、図１７に示す処理にしたがって変換配列中の数値が変化する状態を示す図である。

【図１９】　図１９は、第１の実施の形態において、挿入および削除を順次実行した場合の状態を説明するための図である。

【図２０】　図２０は、第１の実施の形態において、挿入および削除を順次実行した場合の状態を説明するための図である。

【図２１】　図２１は、本発明の第２の実施の形態にかかる変換配列の構成を説明するための図である。

【図２２】　図２２は、本発明の第３の実施の形態にかかる実体配列および差分配列を説明するための図である。

【図２３】　図２３は、本発明の第４の実施の形態にかかる値変換配列の概略を説明するための図である。

【図２４】　図２４は、第４の実施の形態にかかる値変換の処理を示すフローチャートである。

【図２５】　図２５は、第４の実施の形態にかかる値変換テーブルを生成する処理を示すフローチャートである。

【図２６】　図２６は、第４の実施の形態において、実体配列中の値の変換を実行した場合の状態を説明するための図である。

【図２７】　図２７は、本発明において、添え字変換配列および値変換配列を組み合わせたデータ構造の一例を示す図である。

【図２８】　図２８は、本発明の第５の実施の形態にかかる各情報ブロックの構造を示す図である。

【図２９】　図２９は、第５の実施の形態において、特定のレコードを削除する処理を説明するための図である。

【図３０】　図３０は、第５の実施の形態において、特定のレコードを削除する処理を説明するための図である。

【図３１】　図３１は、第５の実施の形態において、特定のレコードを削除する処理を説明するための図である。

【図３２】　図３２は、第５の実施の形態において、特定のレコードを削除する処理を説明するための図である。

【図３３】　図３３は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図３４】　図３４は、第５の実施の形態において、特定のレコードを特定の

位置に挿入する処理を説明するための図である。

【図３５】　図３５は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図３６】　図３６は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図３７】　図３７は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図３８】　図３８は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図３９】　図３９は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図４０】　図４０は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図４１】　図４１は、第５の実施の形態において、特定のレコードを特定の位置に挿入する処理を説明するための図である。

【図４２】　図４２は、本発明の第６の実施の形態において、特定のレコードを削除する処理を説明するための図である。

【図４３】　図４３は、本発明を利用したトランザクション処理を示すフローチャートである。

【図４４】　図４４は、本発明を利用した複数プロセスの並列処理を説明するための図である。

【図４５】　図４５は、本発明を利用した排他処理を説明するための図である。

【図４６】　図４６は、本発明において、情報ブロックの形態をなす表形式データのデータ挿入処理を説明するためのフローチャートである。

【図４７】　図４７は、本発明において、情報ブロックの形態の表形式データのデータ挿入処理を説明するための図である。

【図４８】　図４８は、本発明において、情報ブロックの形態の表形式データのデータ削除処理を説明するための図である。

【図４９】　図４９は、本発明にかかる添え字変換配列の他の応用例を説明するための図である。

【図５０】　図５０は、本発明にかかる値変換配列のさらに他の応用例を説明するための図である。

【図５１】　図５１は、本発明にかかる添え字変換配列および値変換配列のさらに他の応用例を説明するための図である。

【図５２】　図５２は、本発明にかかる情報ブロックの形態の表形式データにおいて、データ挿入処理の他の例を説明するための図である。

【図５３】　図５３は、本発明にかかる情報ブロックの形態の表形式データにおいて、データ更新処理を説明するための図である。

【図５４】　図５４は、本発明にかかる情報ブロックの形態の表形式データにおいて、データ更新処理の他の例を説明するための図である。

【符号の説明】

１０	コンピュータシステム
１２	ＣＰＵ
１４	ＲＡＭ
１６	ＲＯＭ
１８	固定記憶装置
２０	ＣＤ－ＲＯＭドライバ
２２	Ｉ／Ｆ
２４	入力装置
２６	表示装置

【書類名】　図面

【図１】

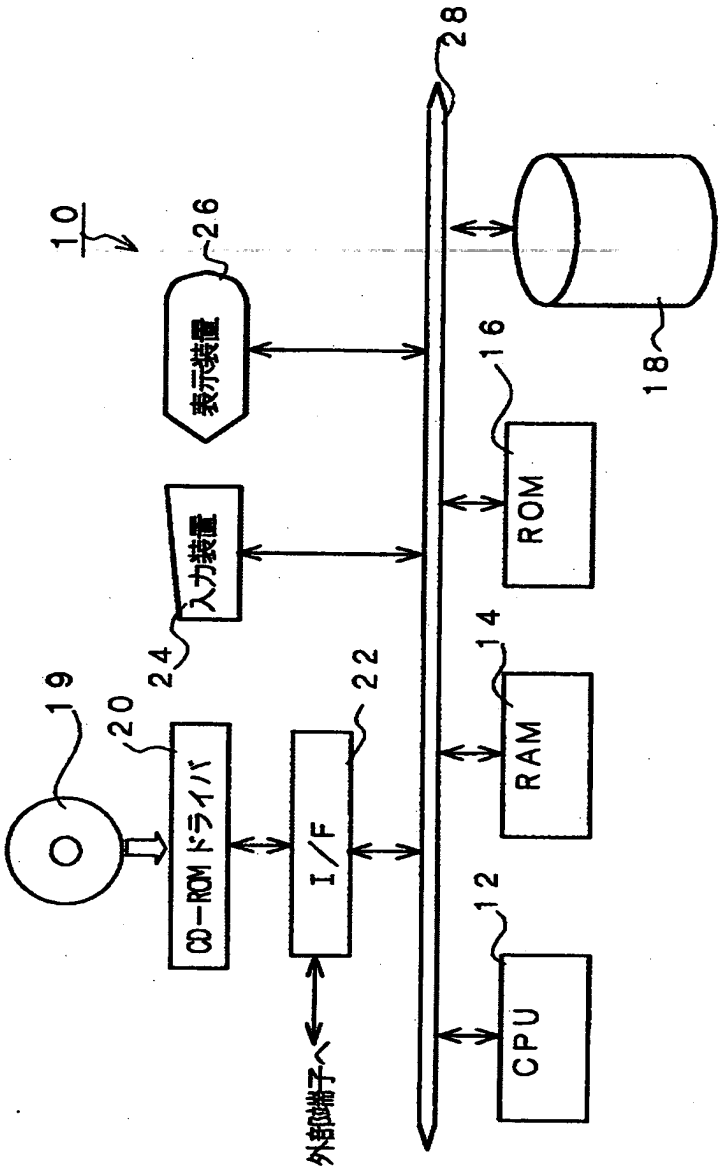
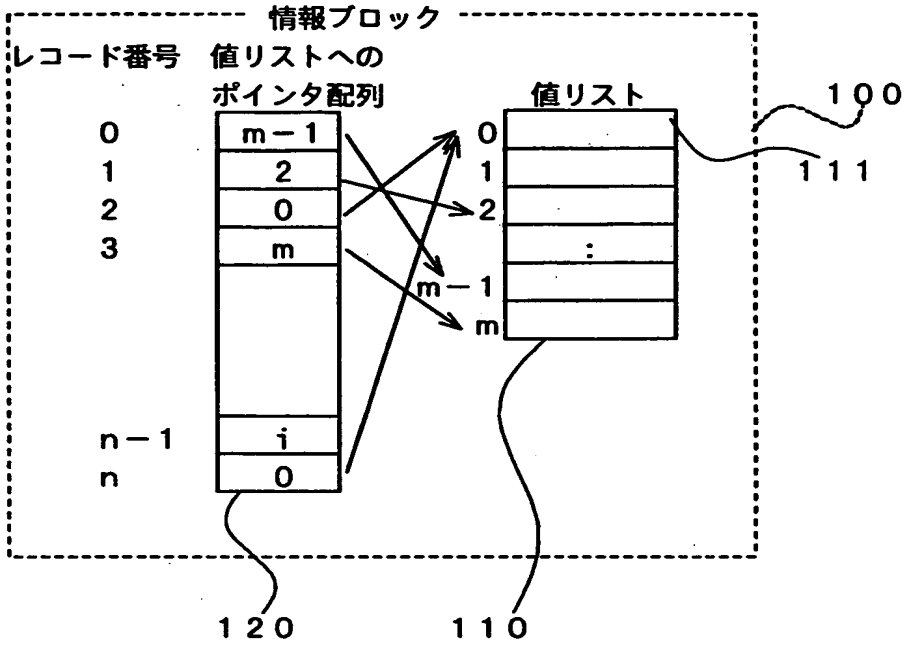

10
26
表示装置
28
18
16
ROM
入力装置
24
19
20
CD-ROMドライバ
22
I/F
14
RAM
12
CPU
外部端子へ

図1

【図２】

図２


情報ブロック
レコード番号
値リストへの
ポインタ配列
値リスト
100
111
0
1
2
3
n－1
n
m－1
2
0
m
i
0
0
1
2
:
m－1
m
120
110

【図３】

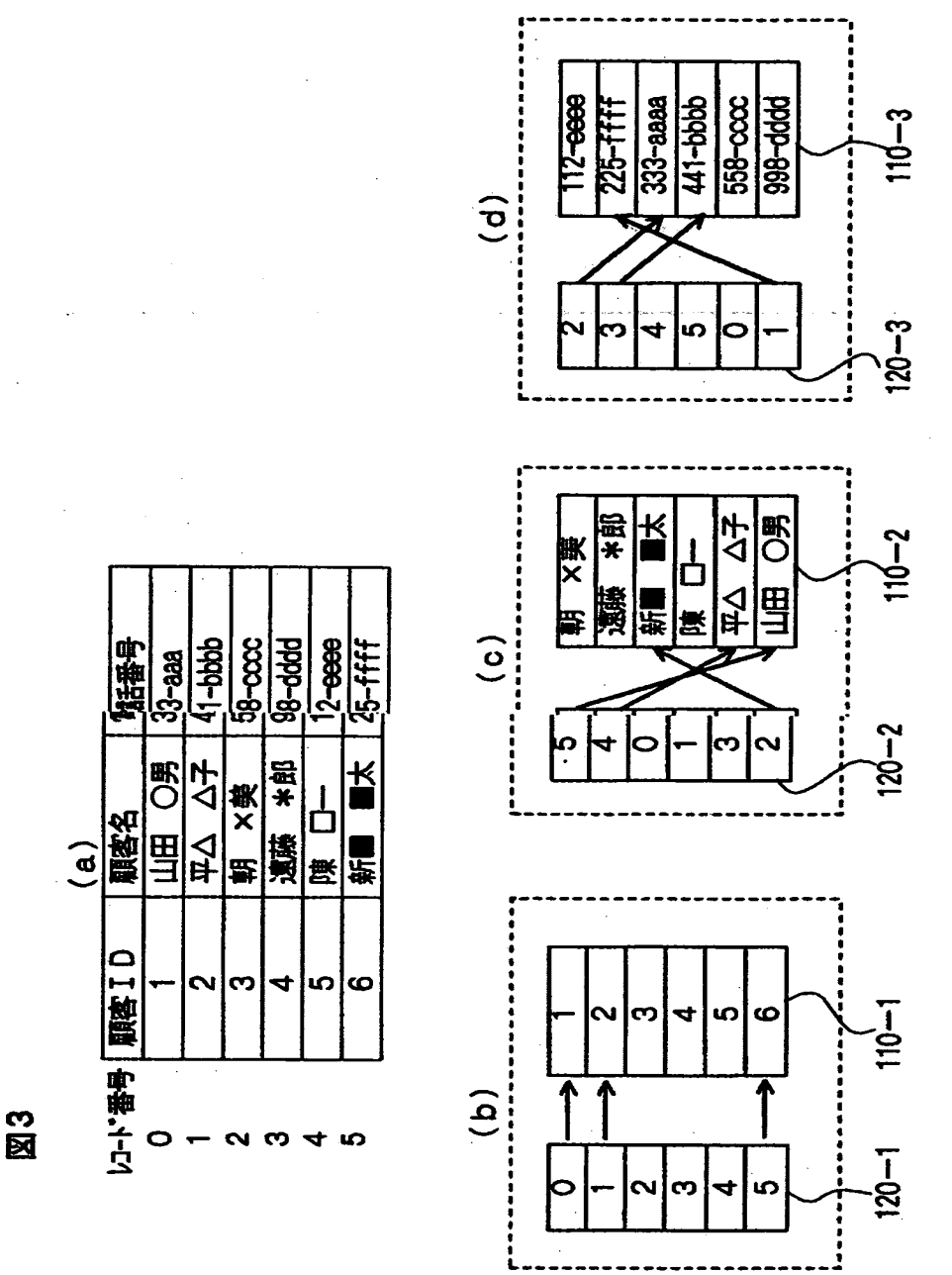

(a)
レコード番号
顧客ID
顧客名
電話番号
0 1 山田 ○男 33-aaa
1 2 平△ △子 41-bbbb
2 3 朝 ×美 58-cccc
3 4 遠藤 *郎 98-dddd
4 5 陳 □一 12-eeee
5 6 新■ ■太 25-ffff
(b)
0 1 2 3 4 5
1 2 3 4 5 6
120-1
110-1
(c)
5 4 0 1 3 2
朝 ×美
遠藤 *郎
新■ ■太
陳 □一
平△ △子
山田 ○男
120-2
110-2
(d)
2 3 4 5 0 1
112-eeee
225-ffff
333-aaaa
441-bbbb
558-cccc
998-dddd
120-3
110-3

図3

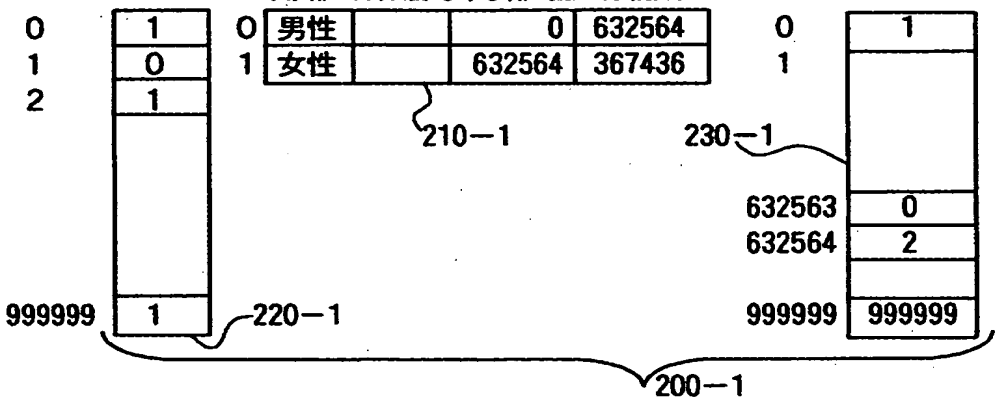
【図４】

図４

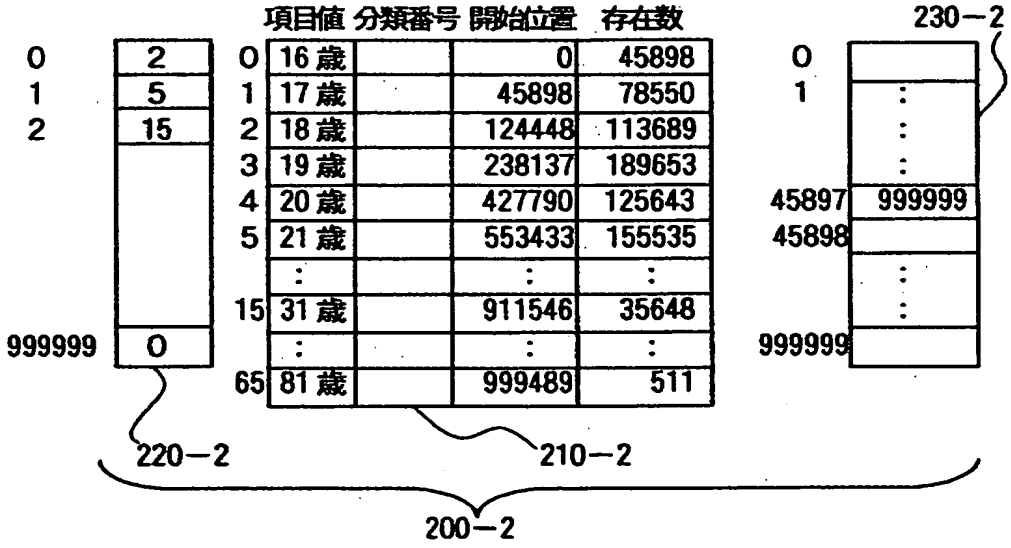
(a)

レコード番号	性別	年令	職業
0	女性	１８	プログラマ
1	男性	２１	学生
2	女性	３１	教員
：	：	：	：
：	：	：	：
999999	女性	１６	学生

(b)

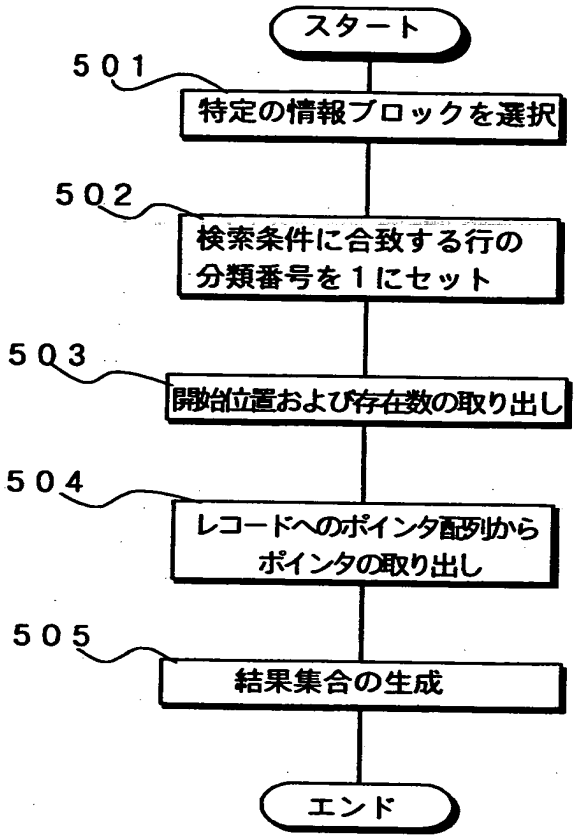


(c)



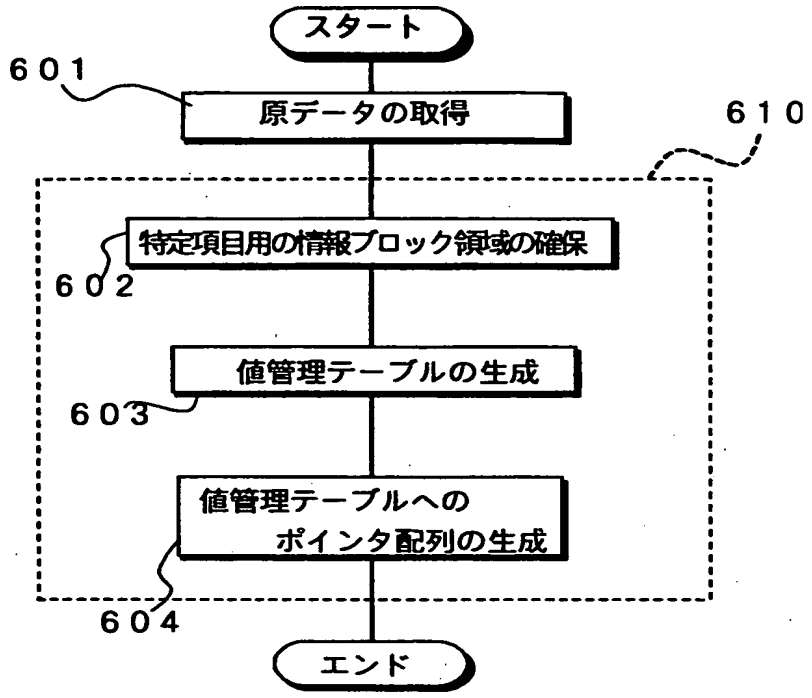
【図５】

図５


スタート
５０１
特定の情報ブロックを選択
５０２
検索条件に合致する行の
分類番号を１にセット
５０３
開始位置および存在数の取り出し
５０４
レコードへのポインタ配列から
ポインタの取り出し
５０５
結果集合の生成
エンド

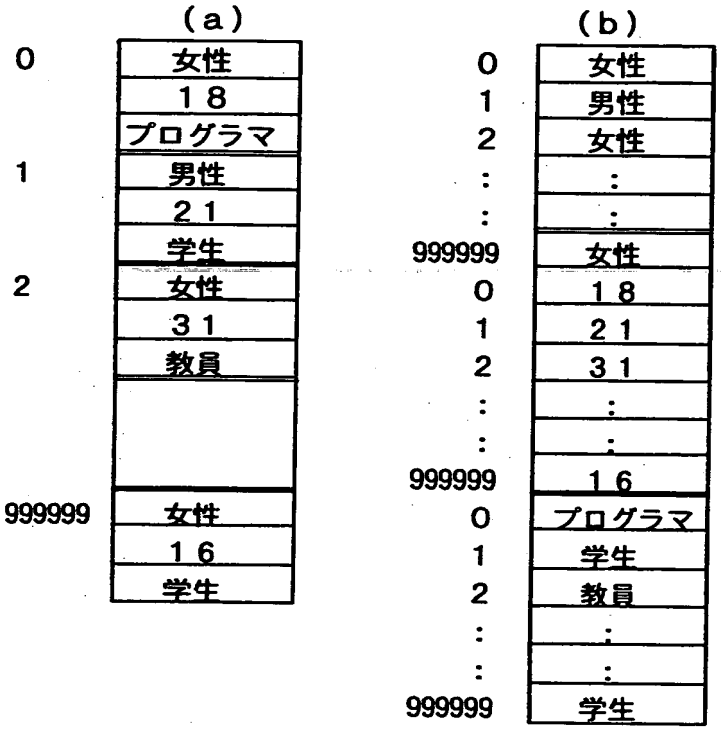
【図６】

図６


スタート
６０１
原データの取得
６１０
特定項目用の情報ブロック領域の確保
６０２
値管理テーブルの生成
６０３
値管理テーブルへの
ポインタ配列の生成
６０４
エンド

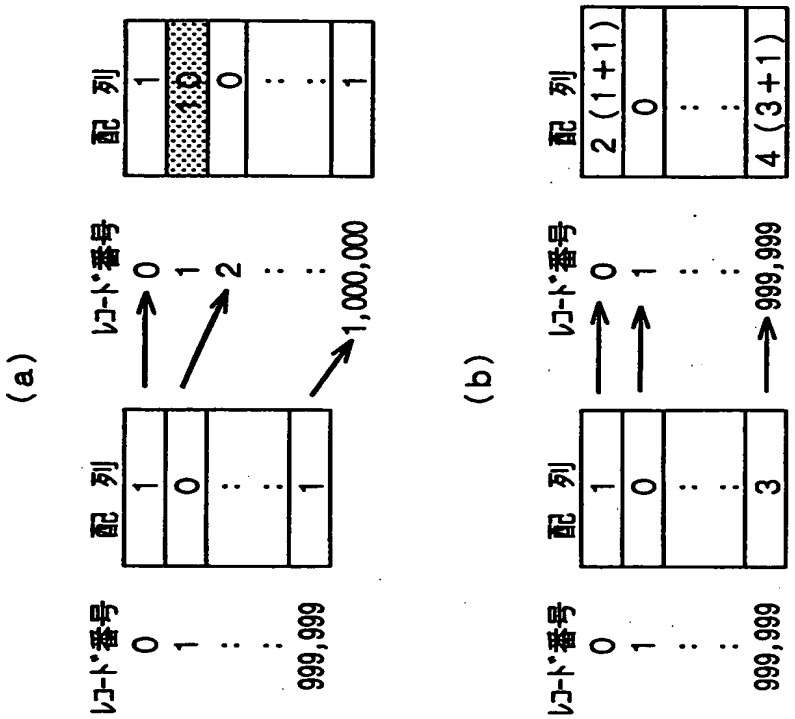
【図７】

図７


(a)
0
女性
１８
プログラマ
1
男性
２１
学生
2
女性
３１
教員
999999
女性
１６
学生
(b)
0
女性
1
男性
2
女性
999999
女性
0
１８
1
２１
2
３１
999999
１６
0
プログラマ
1
学生
2
教員
999999
学生

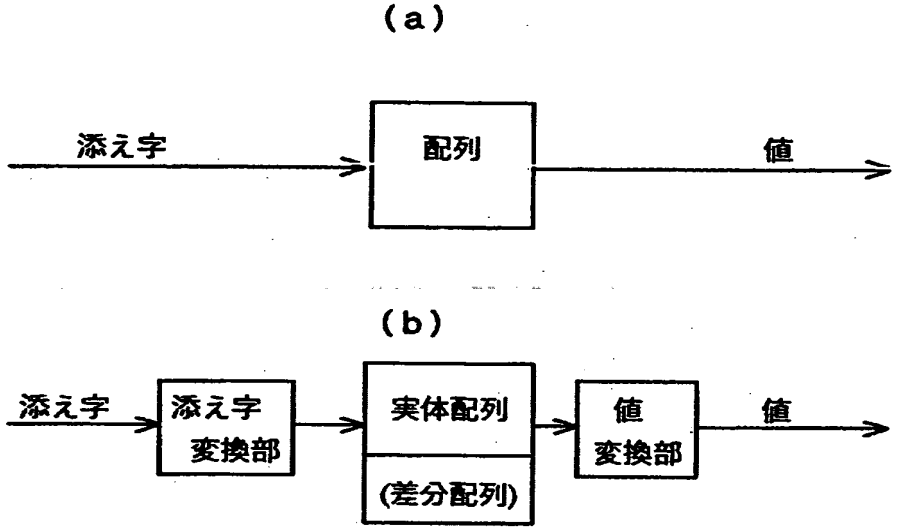
【図８】

図8


(a)
レコード番号
0
1
2
:
:
1,000,000
配列
1
1.5
0
:
:
1
レコード番号
0
1
:
:
999,999
配列
1
0
:
:
1
(b)
レコード番号
0
1
:
:
999,999
配列
2 (1+1)
0
:
:
4 (3+1)
レコード番号
0
1
:
:
999,999
配列
1
0
:
:
3

【図９】

図９


(a)
添え字
配列
値
(b)
添え字
添え字
変換部
実体配列
(差分配列)
値
変換部
値
添え字および値の双方を変換する例
(c)
添え字
添え字
変換部
実体配列
(差分配列)
値
添え字のみを変換する例
(d)
添え字
実体配列
(差分配列)
値
変換部
値
値のみを変換する例

【図１０】

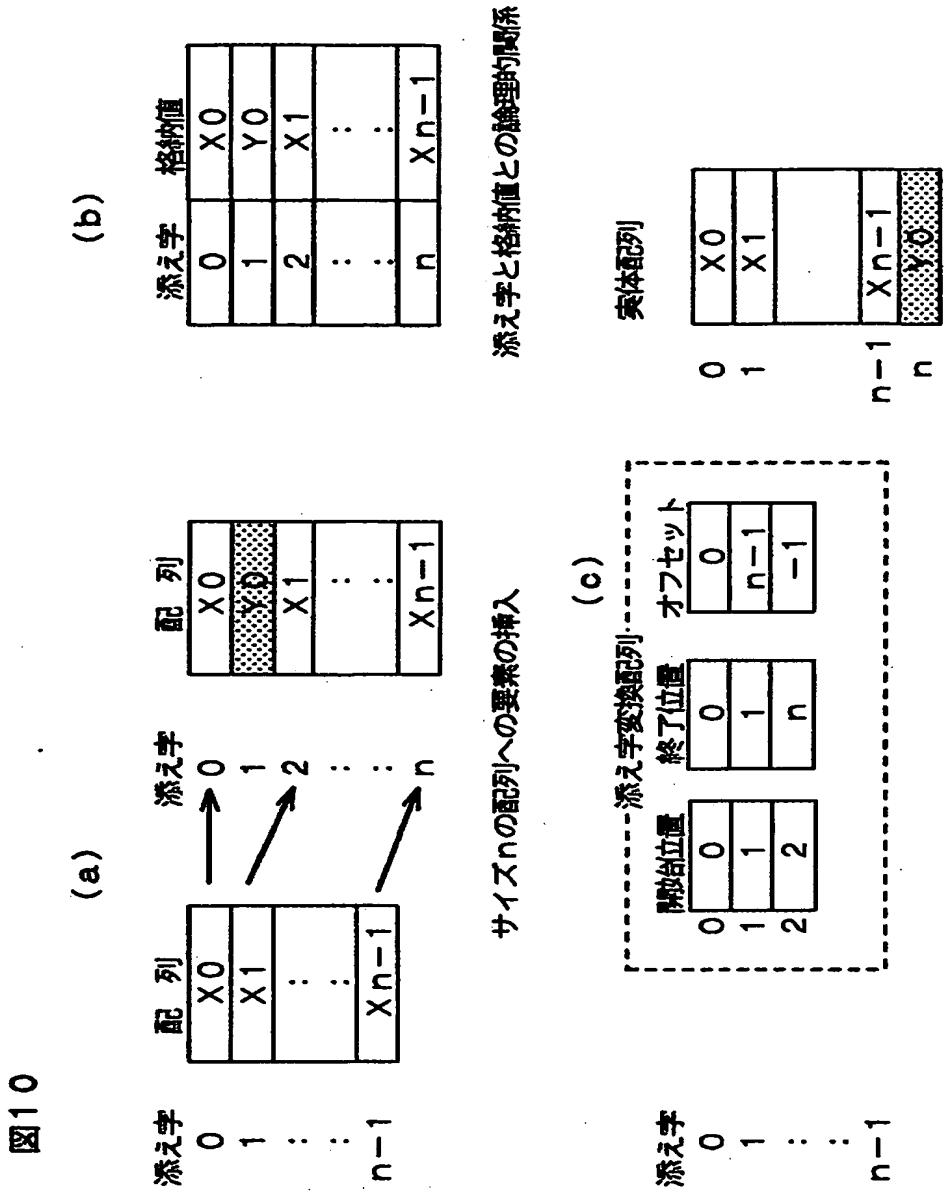
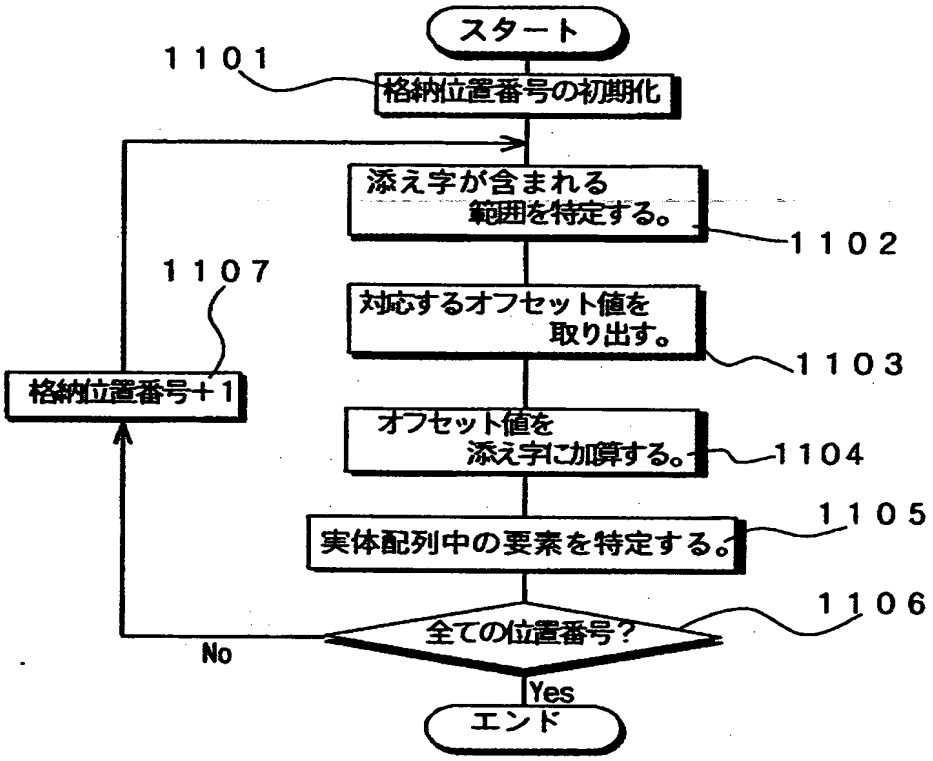


図１０
(a)
添え字
配 列
X0
X1
Xn−1
0
1
n−1
サイズｎの配列への要素の挿入
(b)
添え字と格納値との論理的関係
添え字
格納値
0
1
2
n
X0
Y0
X1
Xn−1
(c)
添え字変換配列
開始位置
終了位置
オフセット
実体配列
Y0

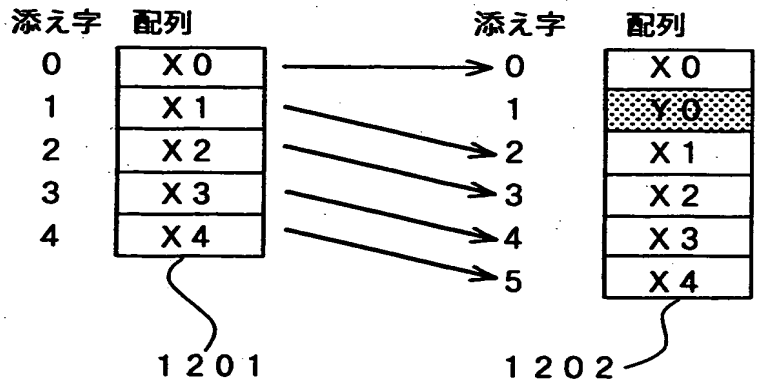
【図１１】

図１１　　（ａ）



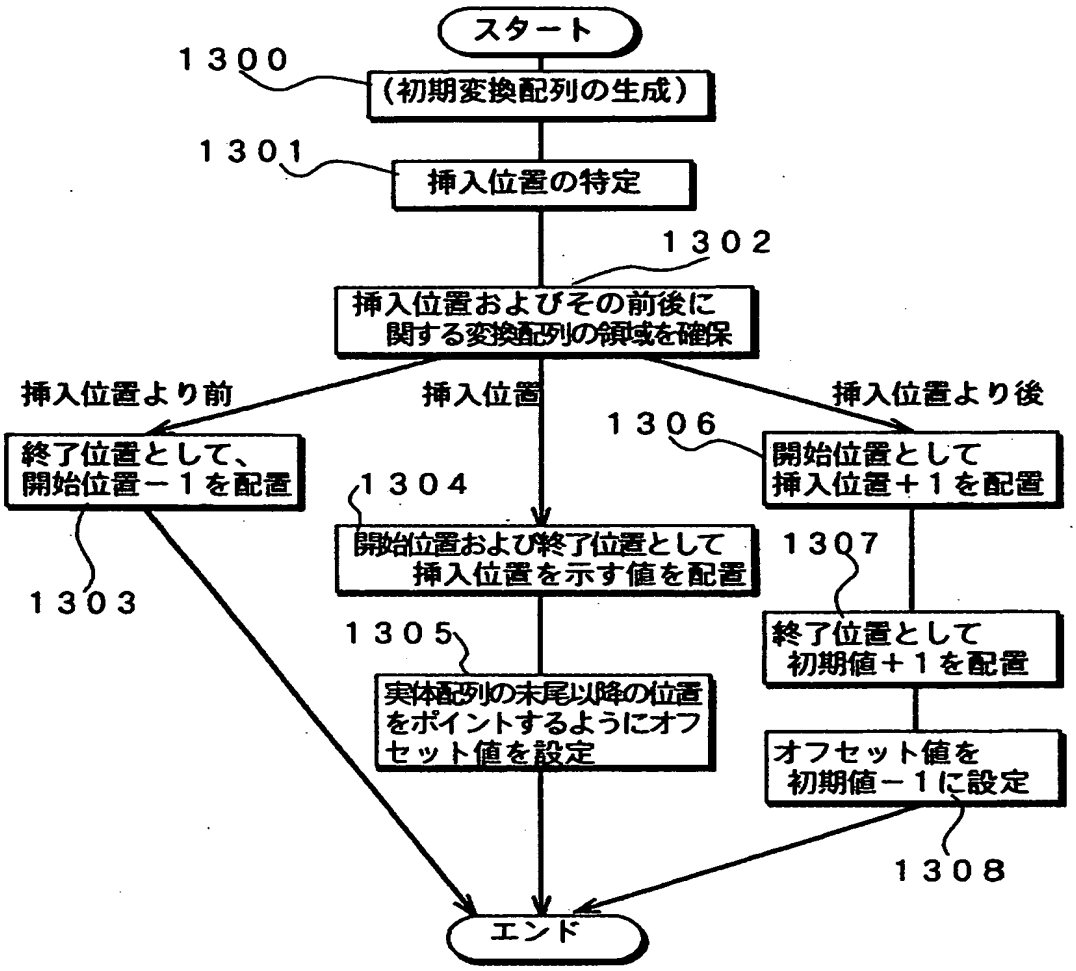
【図１２】

図１２



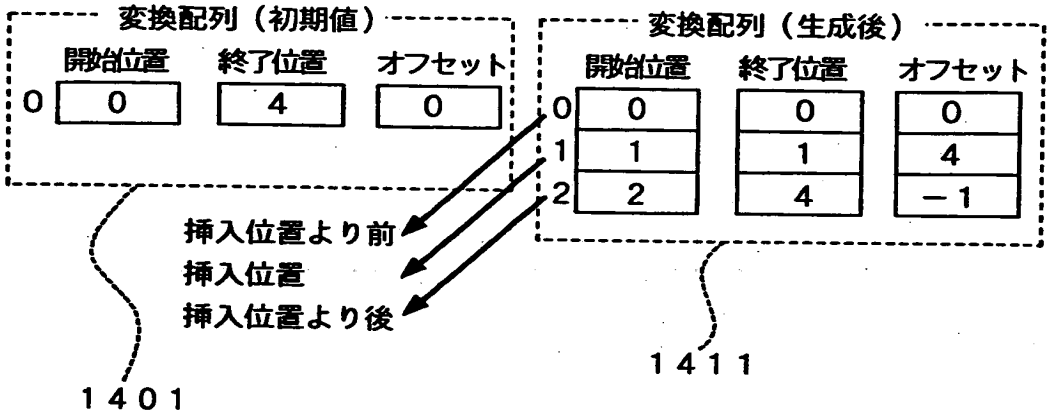
【図１３】

図１３


スタート
１３００
（初期変換配列の生成）
１３０１
挿入位置の特定
１３０２
挿入位置およびその前後に
関する変換配列の領域を確保
挿入位置より前
挿入位置
挿入位置より後
１３０６
終了位置として、
開始位置－１を配置
１３０４
開始位置として
挿入位置＋１を配置
開始位置および終了位置として
挿入位置を示す値を配置
１３０７
１３０３
１３０５
終了位置として
初期値＋１を配置
実体配列の末尾以降の位置
をポイントするようにオフ
セット値を設定
オフセット値を
初期値－１に設定
１３０８
エンド

【図１４】

図１４


変換配列（初期値）
開始位置
終了位置
オフセット
0
0
4
0
変換配列（生成後）
開始位置
終了位置
オフセット
0
0
0
0
1
1
1
4
2
2
4
−1
挿入位置より前
挿入位置
挿入位置より後
１４０１
１４１１

【図１５】

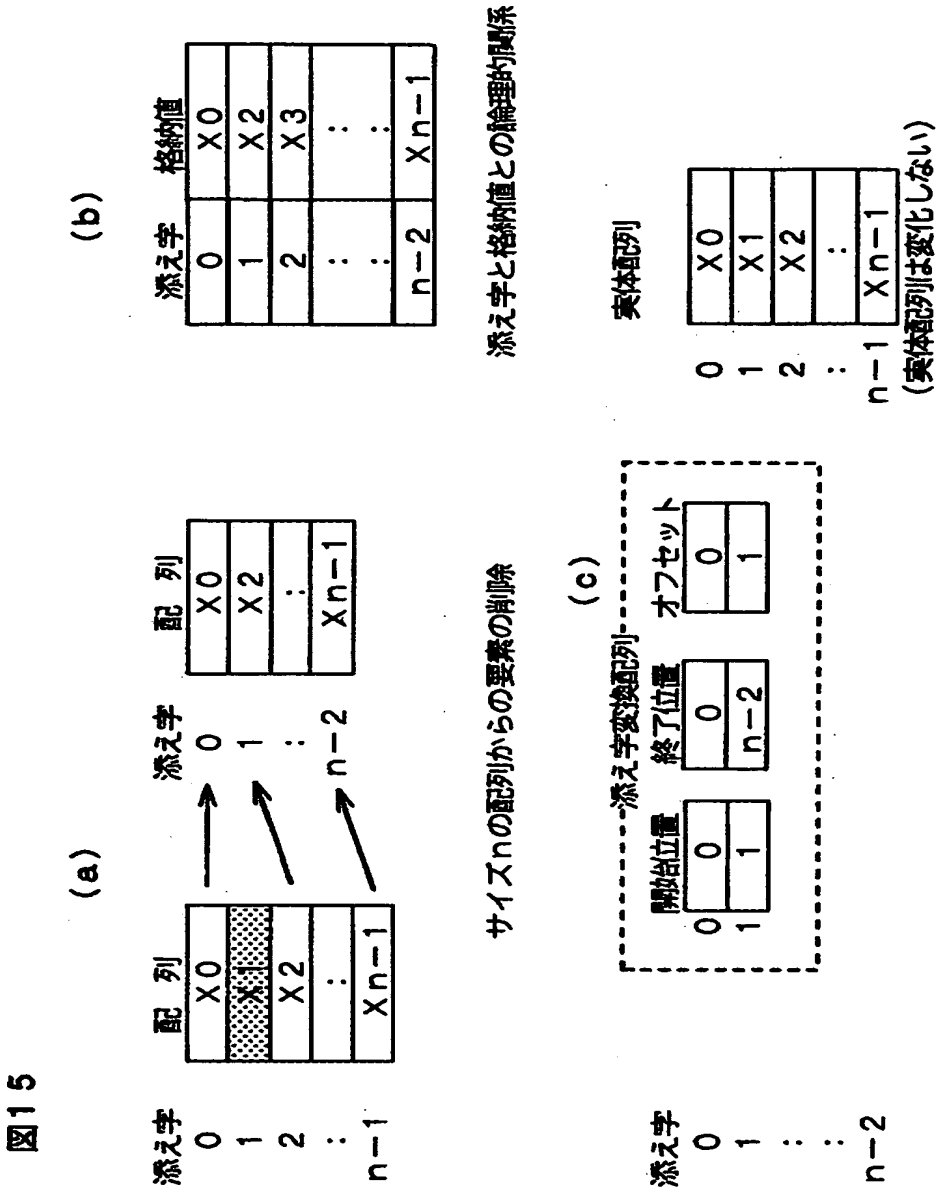
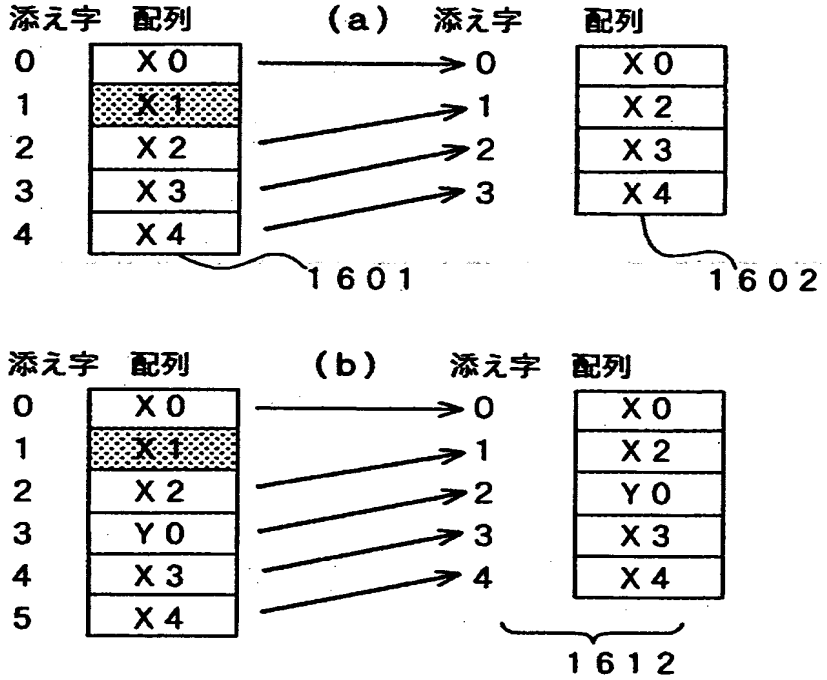


図15
(a)
添え字
配 列
0
1
2
n−1
X0
X1
X2
Xn−1
n−2
サイズnの配列からの要素の削除
(b)
添え字と格納値との論理的関係
添え字
格納値
X0
X2
X3
Xn−1
実体配列
(実体配列は変化しない)
(c)
添え字変換配列
開始位置
終了位置
オフセット

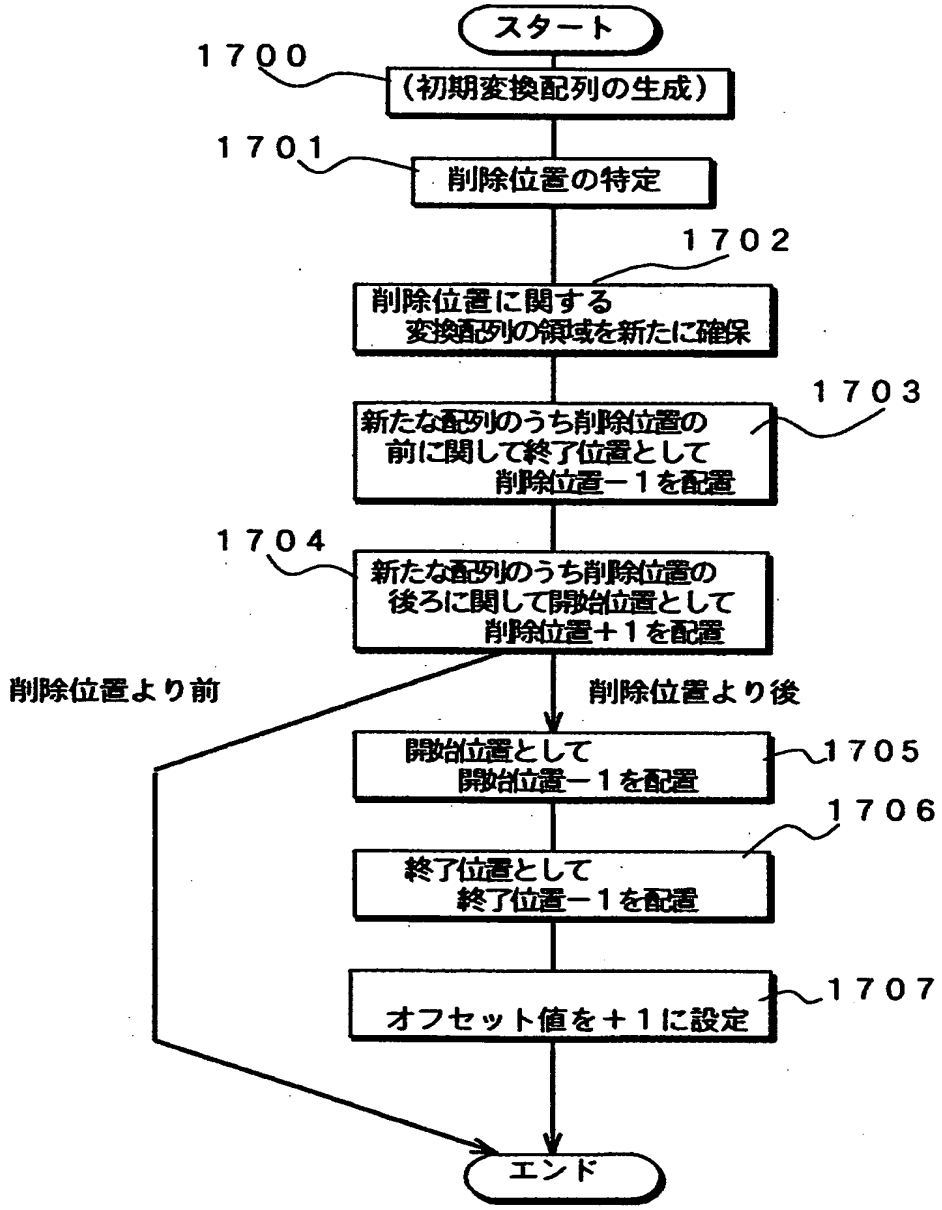
【図１６】

図１６


添え字
配列
(a)
添え字
配列
0
1
2
3
4
X０
X１
X２
X３
X４
０
１
２
３
X０
X２
X３
X４
１６０１
１６０２
添え字
配列
(b)
添え字
配列
0
1
2
3
4
5
X０
X１
X２
Y０
X３
X４
０
１
２
３
４
X０
X２
Y０
X３
X４
１６１２
実体配列
0
1
2
3
4
5
XO
X１
X２
X３
X４
Y０
１６１０

【図１７】

図１７


スタート
１７００
（初期変換配列の生成）
１７０１
削除位置の特定
１７０２
削除位置に関する
変換配列の領域を新たに確保
１７０３
新たな配列のうち削除位置の
前に関して終了位置として
削除位置－１を配置
１７０４
新たな配列のうち削除位置の
後ろに関して開始位置として
削除位置＋１を配置
削除位置より前
削除位置より後
開始位置として
開始位置－１を配置
１７０５
１７０６
終了位置として
終了位置－１を配置
１７０７
オフセット値を＋１に設定
エンド

【図１８】

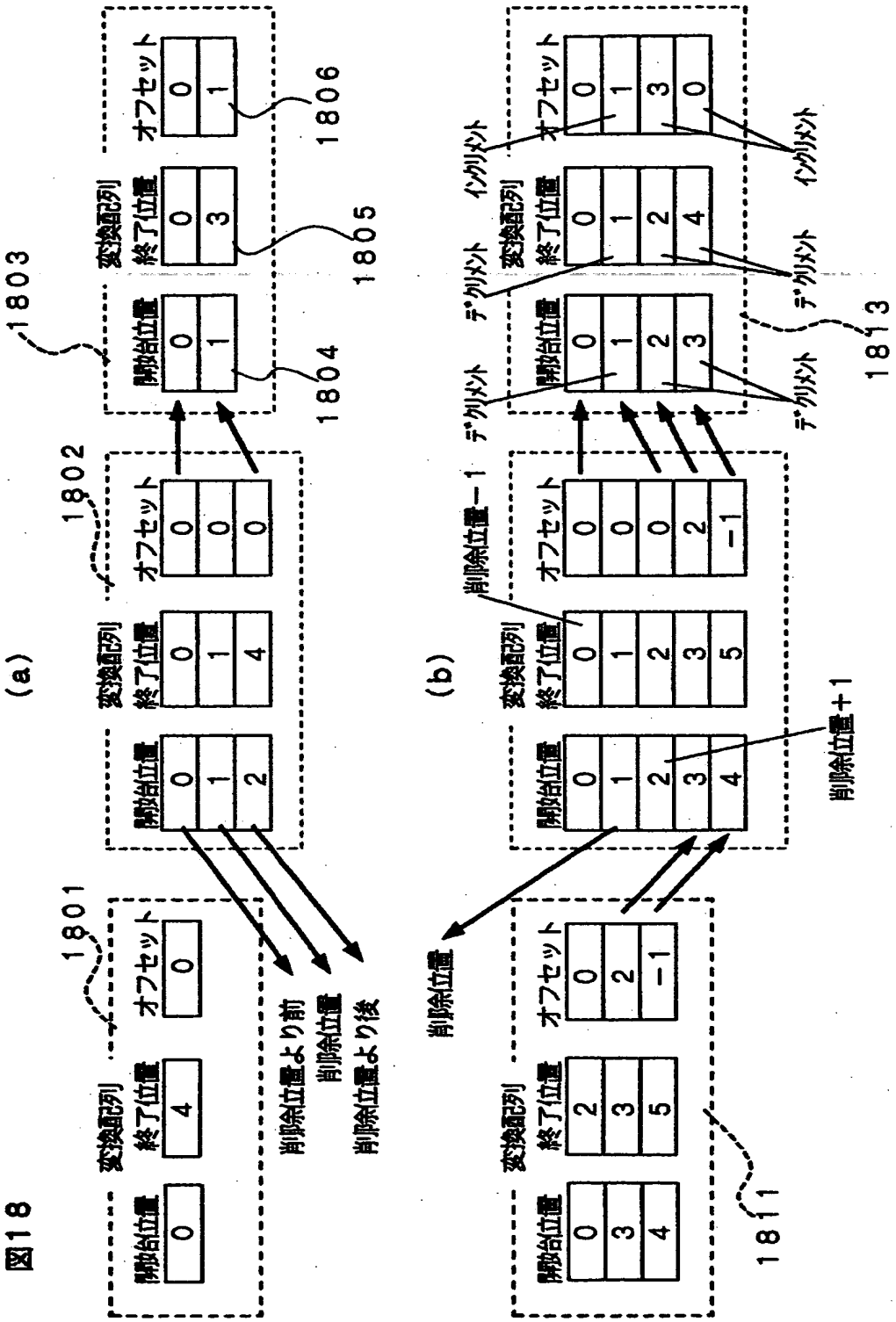
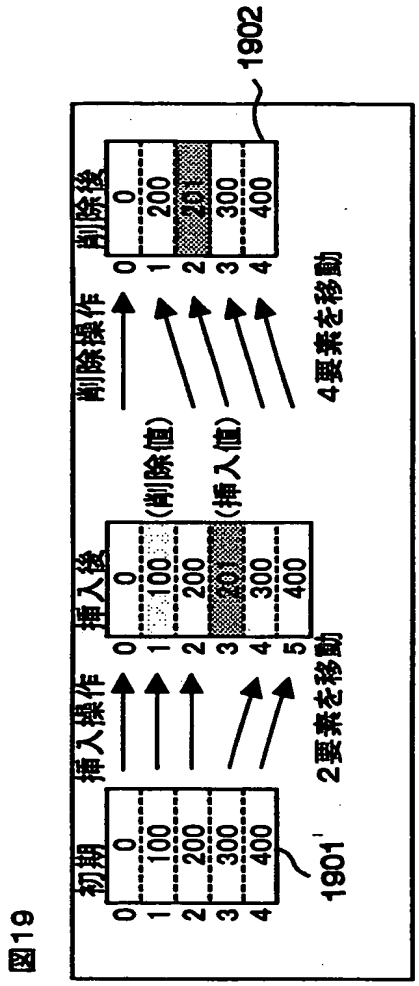


図18
(a)
(b)
1801
1802
1803
1804
1805
1806
1811
1813
開始位置
終了位置
変換配列
オフセット
削除余位置より前
削除余位置
削除余位置より後
削除余位置
削除余位置－１
削除余位置＋１
デクリメント
インクリメント

【図１９】


1902
削除後
0
200
201
300
400
0 1 2 3 4
削除操作
4要素を移動
(削除値)
(挿入値)
挿入後
0
100
200
201
300
400
0 1 2 3 4 5
挿入操作
2要素を移動
初期
0
100
200
300
400
0 1 2 3 4
1901
図19

【図２０】

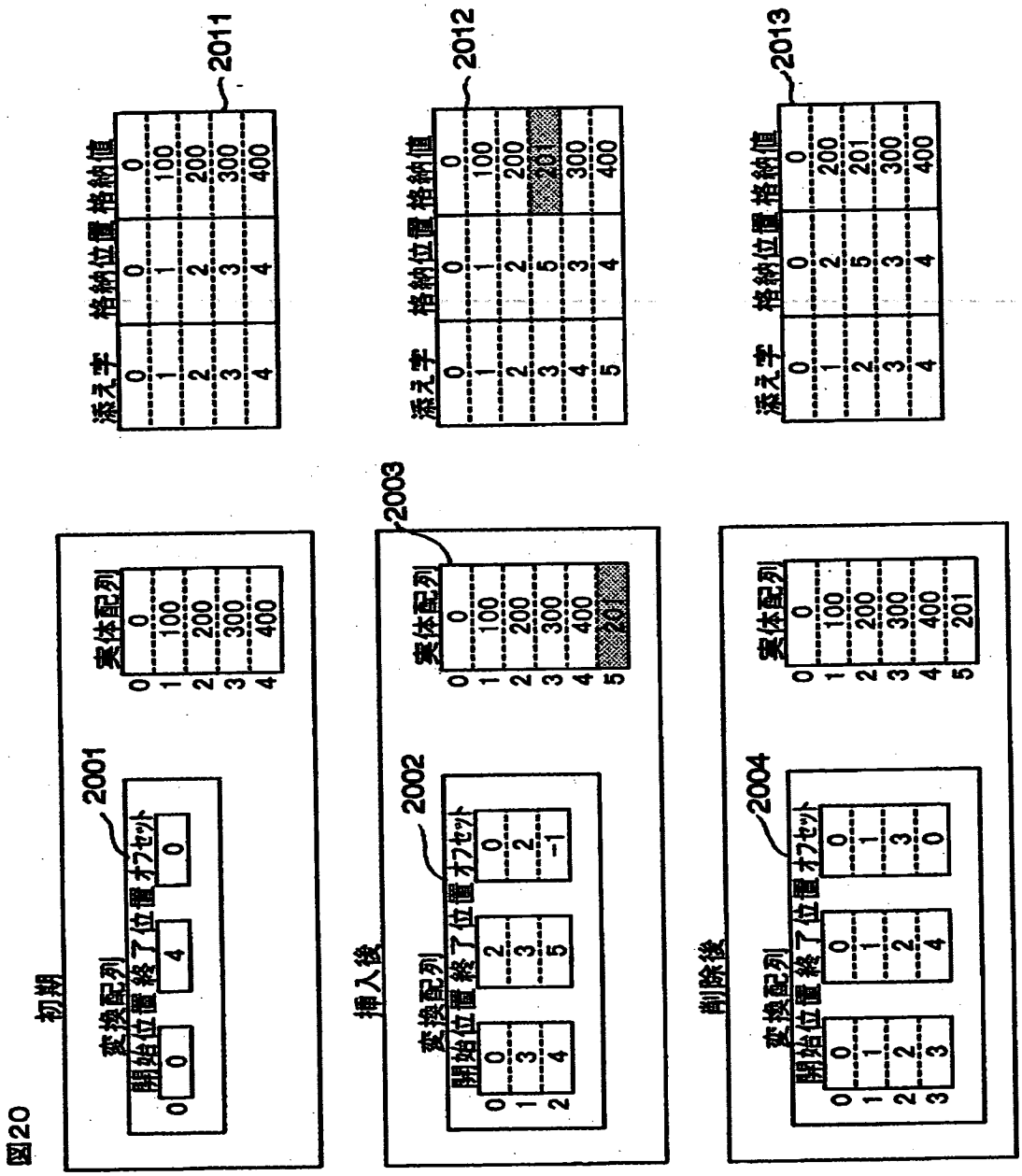
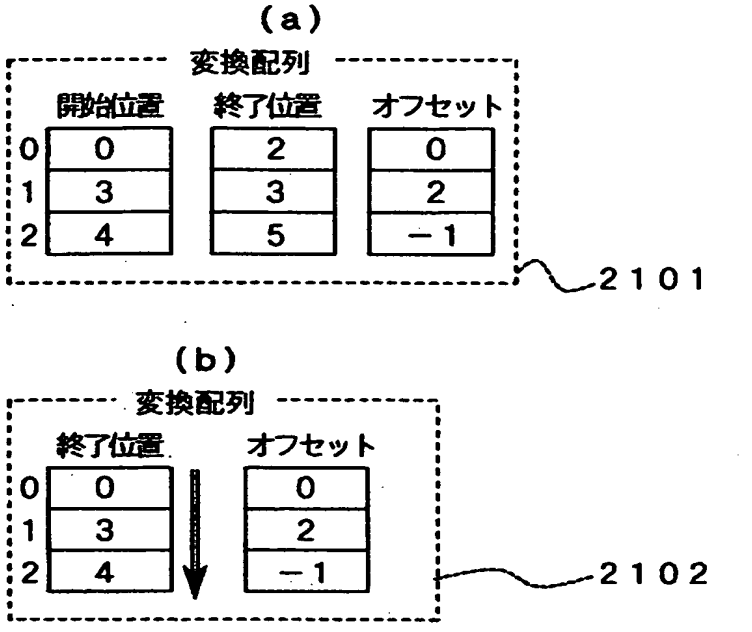


図20
初期
2001
変換配列
開始位置
終了位置
オフセット
実体配列
格納位置
格納値
挿入後
2002
2003
削除後
2004
2011
2012
2013
添え字

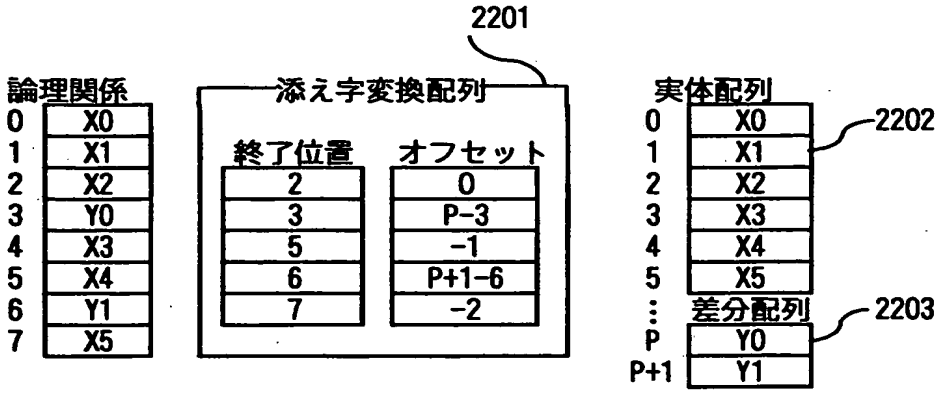
【図２１】

図２１



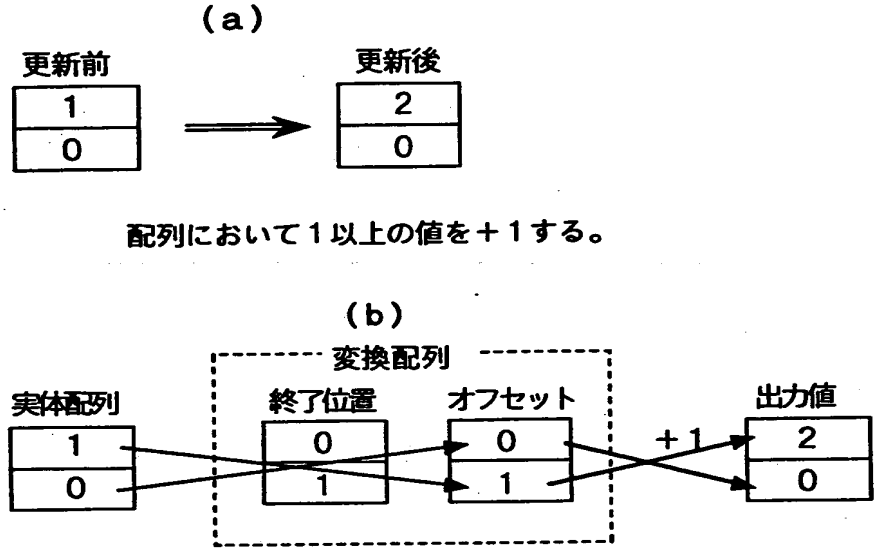
【図２２】

図２２



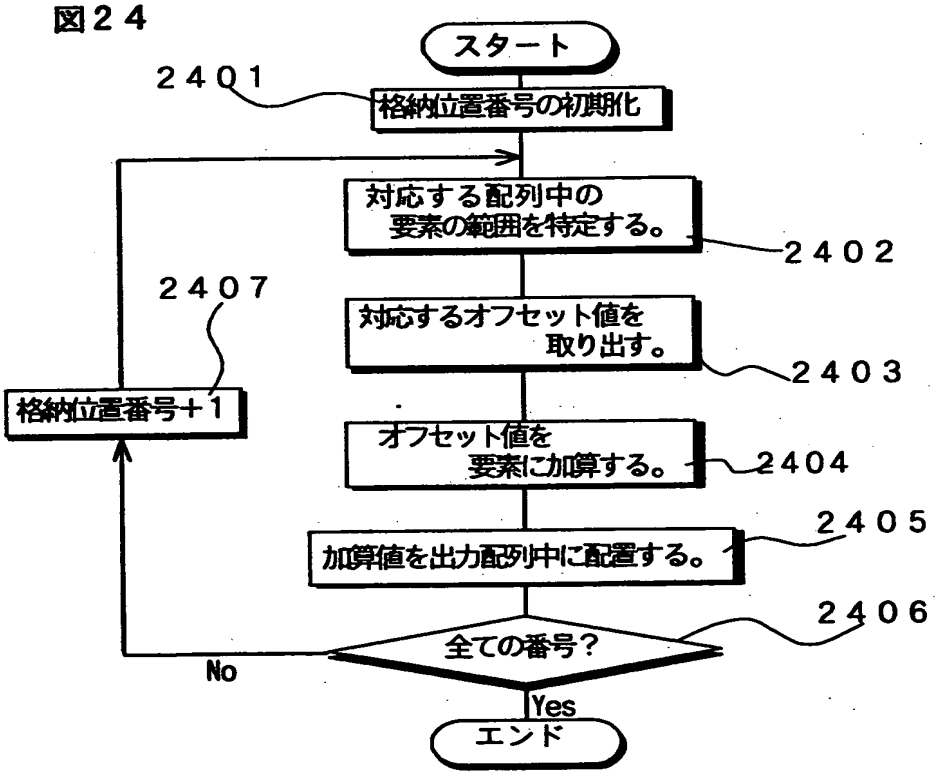
【図２３】

図２３



【図２４】

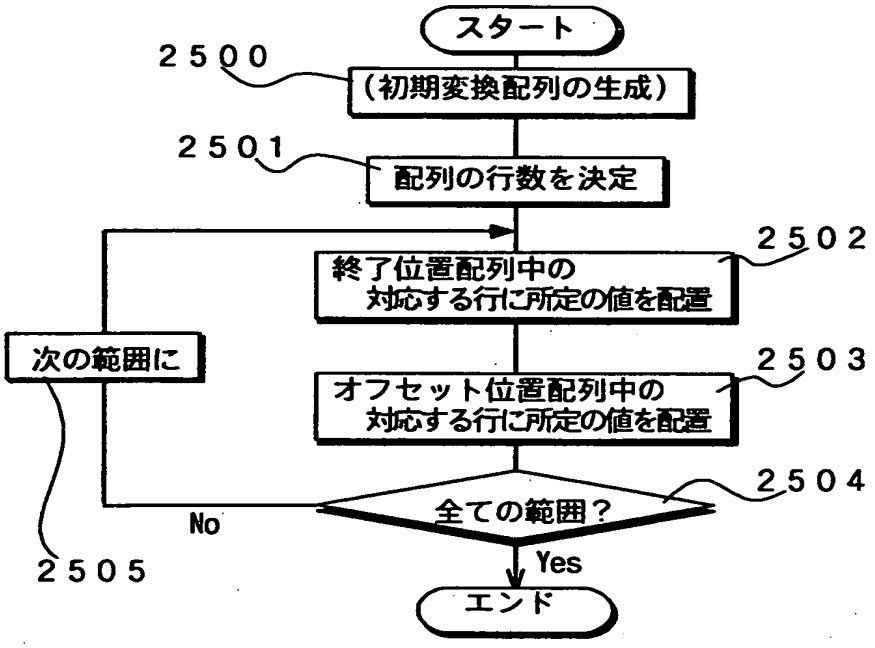
図２４



出証特２０００－３０６９０３５

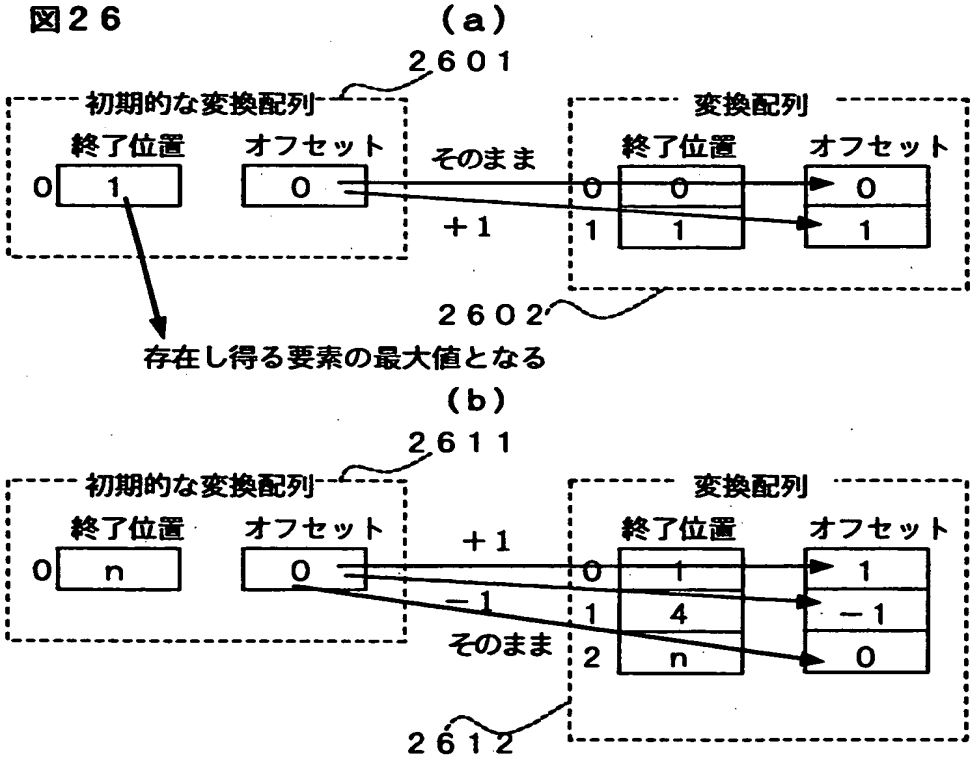
【図２５】

図２５



【図２６】

図２６



【図２７】

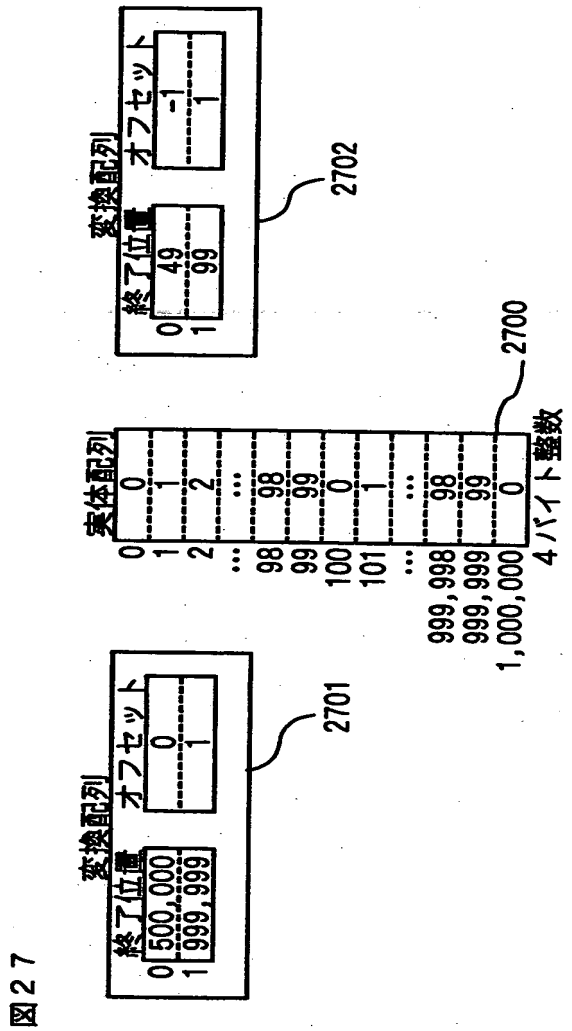
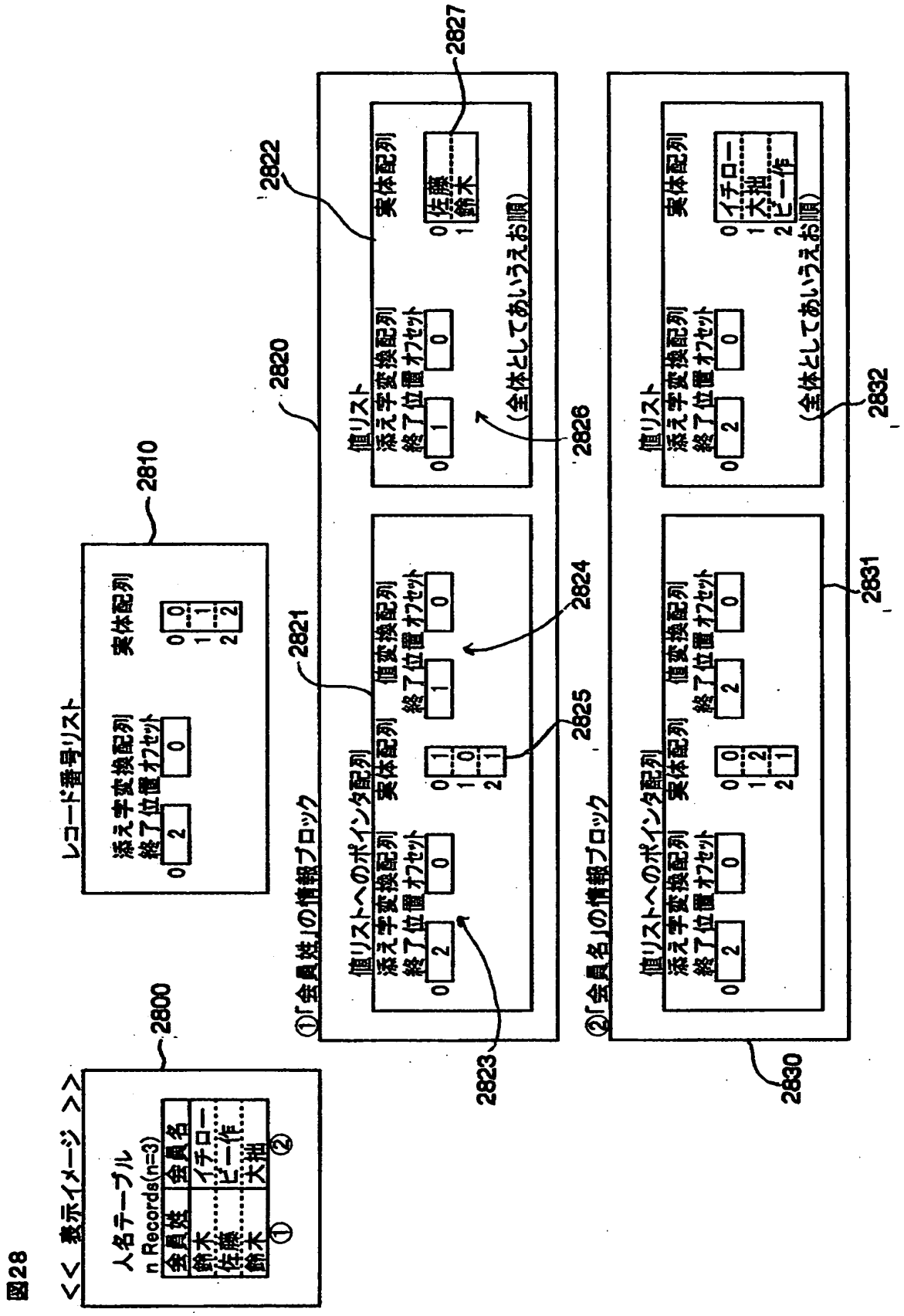

変換配列
終了位置
オフセット
0
1
500,000
999,999
-1
1
2701
2702
実体配列
4バイト整数
2700
0
1
2
...
98
99
100
101
...
999,998
999,999
1,000,000

図２７

【図２８】

図28


<< 表示イメージ >>
人名テーブル
n Records(n=3)
全員姓
全員名
鈴木
イチロー
佐藤
ピーター
鈴木
大地
①
②
2800
レコード番号リスト
添え字変換配列
終了位置オフセット
実体配列
2810
①「全員姓」の情報ブロック
2820
値リスト
2821
値変換配列
2822
2823
2824
2825
2826
2827
(全体としてあいうえお順)
②「全員名」の情報ブロック
2830
2831
2832

【図２９】

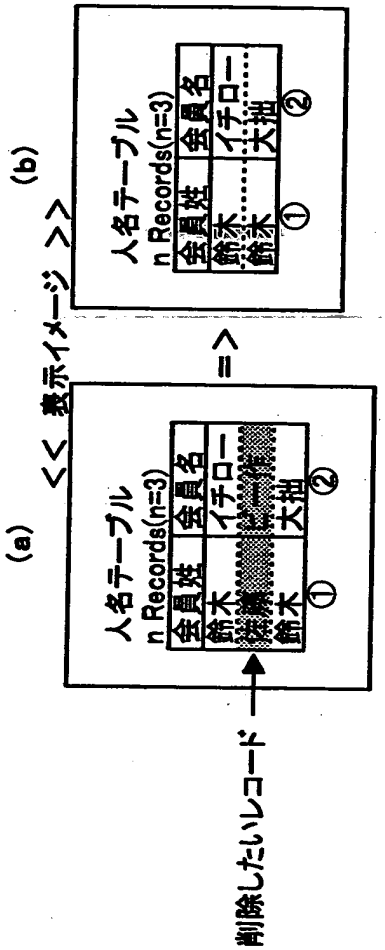

(a)
<< 表示イメージ >>
(b)
=>
人名テーブル
n Records(n=3)
全員姓
全員名
鈴木
イチロー
鈴木
大地
①
②
削除したいレコード

図29

【図３０】

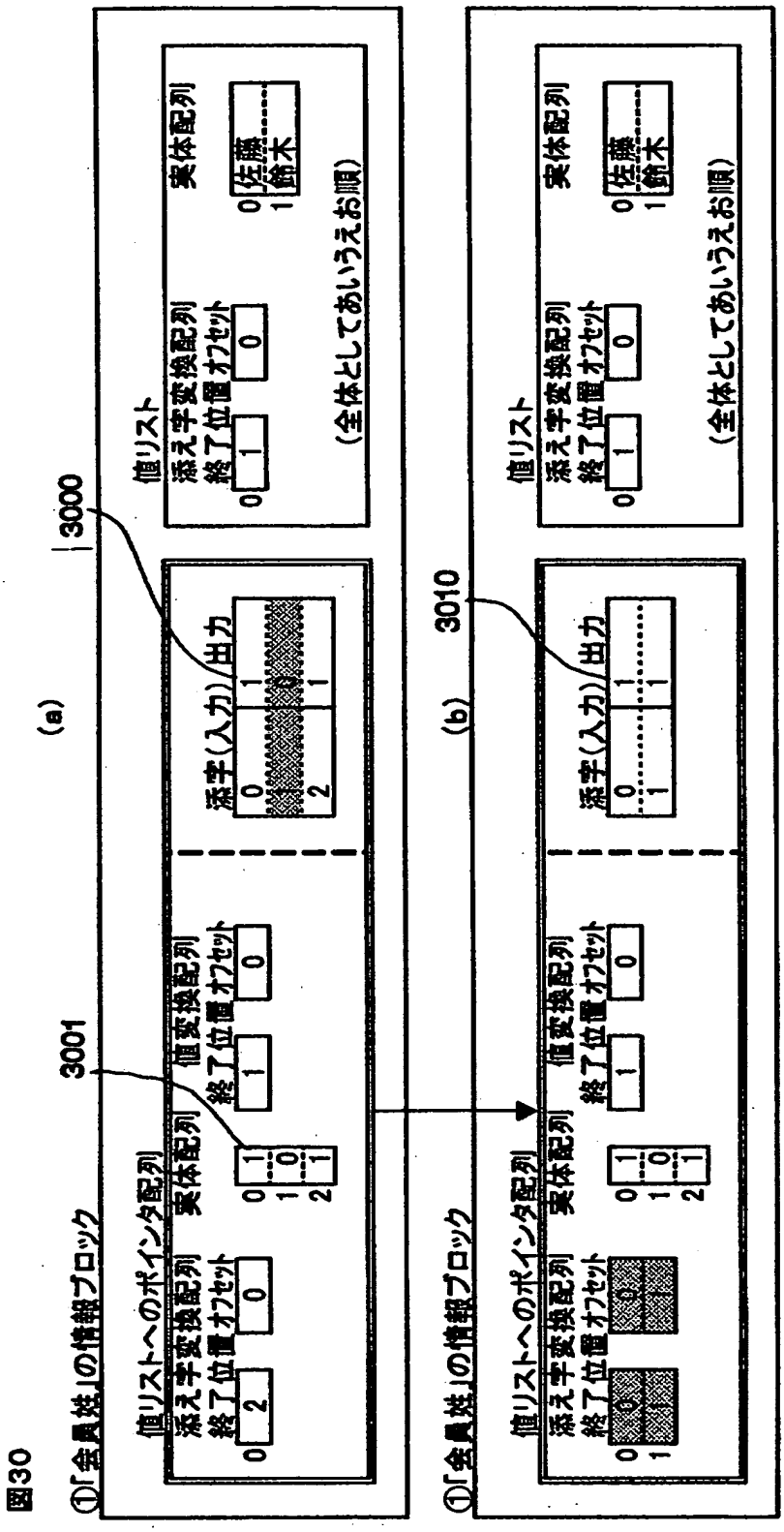


図30
①「金属姓」の情報ブロック
(a)
3000
3001
値リストへのポインタ配列
実体配列
添え字変換配列
終了位置オフセット
添字(入力)/出力
値リスト
(全体としてあいうえお順)
佐藤
鈴木
①「金属姓」の情報ブロック
(b)
3010

【図３１】

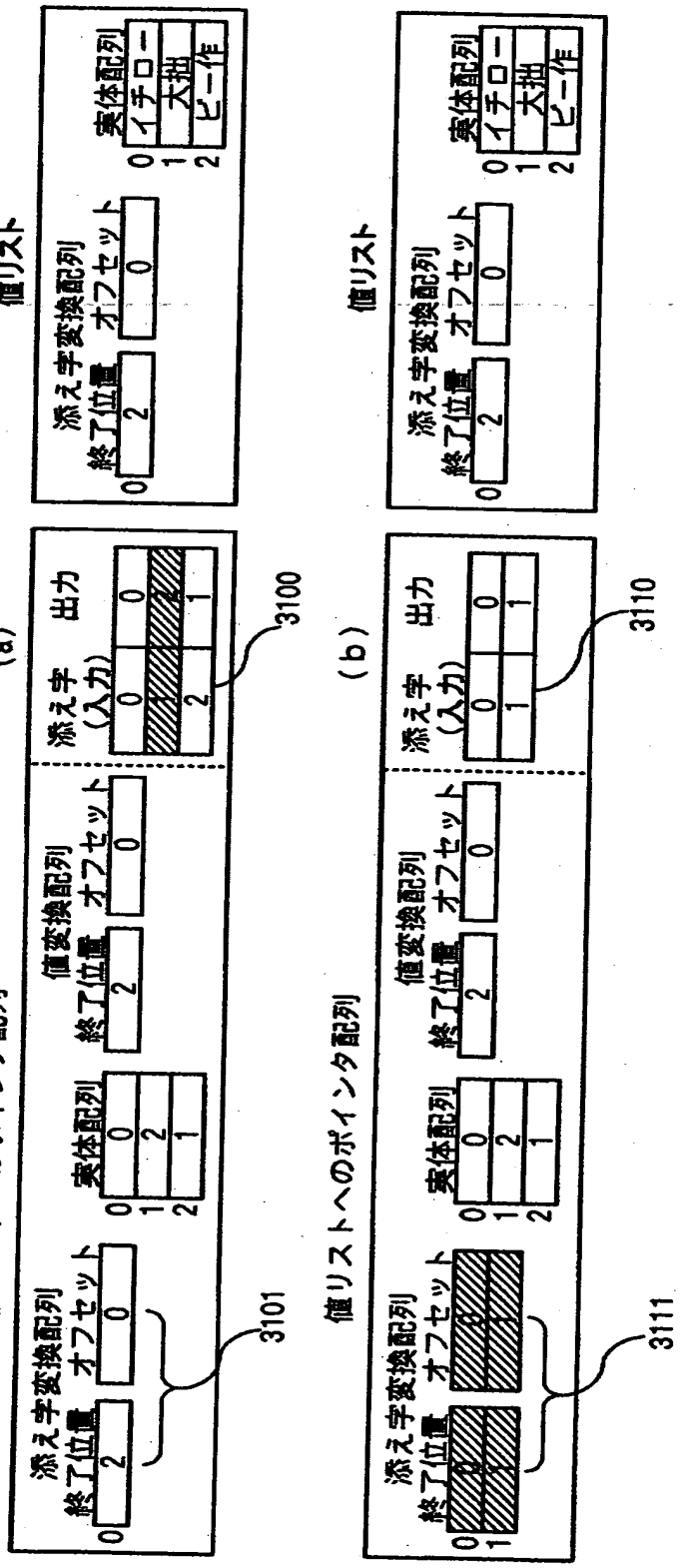


図３１
(a)
値リスト
添え字変換配列
終了位置
オフセット
実体配列
イチロー
大拙
ピーター
値リストへのポインタ配列
値変換配列
添え字
(入力)
出力
3100
3101
(b)
3110
3111

【図３２】

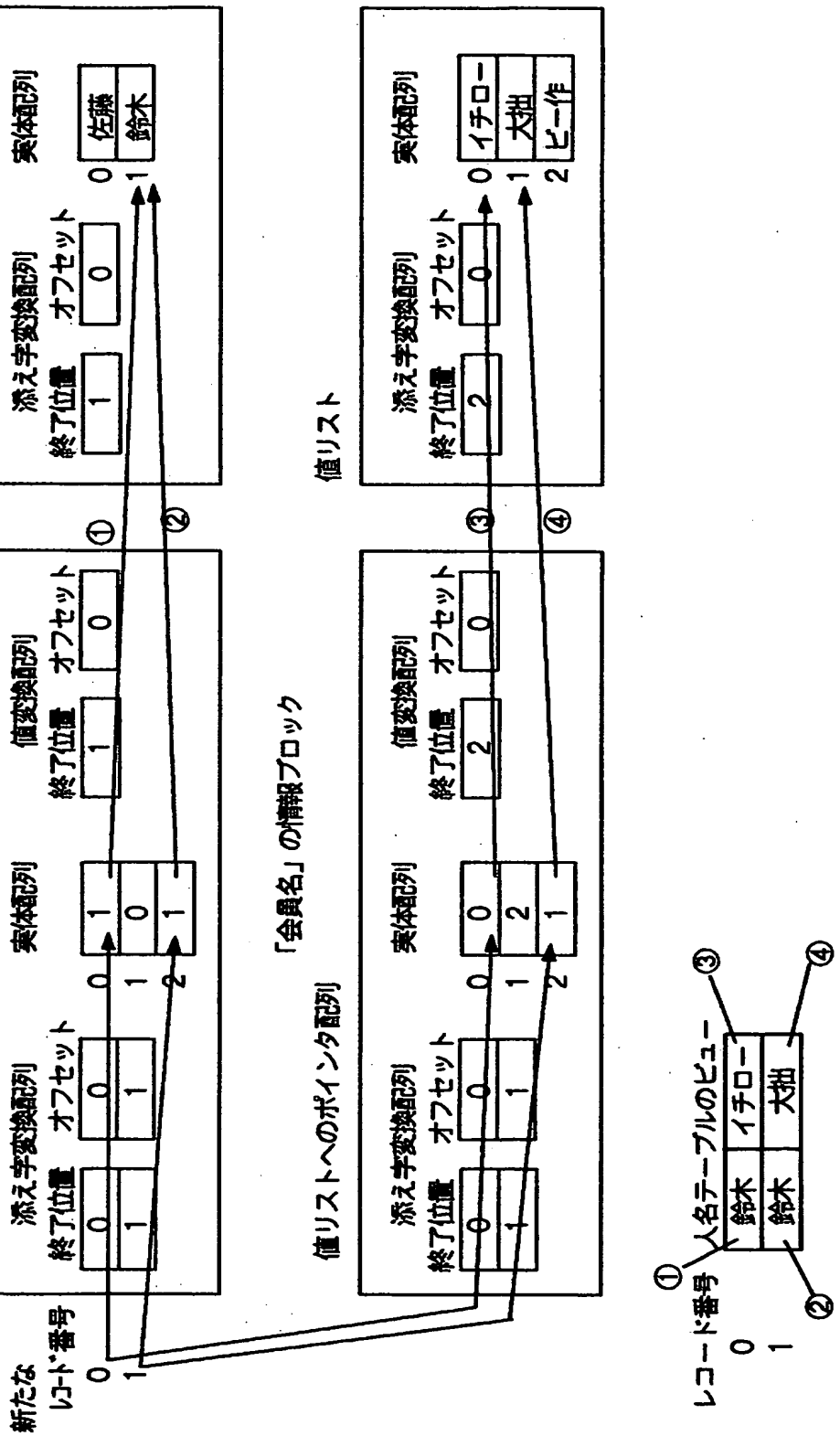


図３２
「会員姓」の情報ブロック
「会員名」の情報ブロック
値リストへのポインタ配列
値リスト
添え字変換配列
終了位置
オフセット
値変換配列
実体配列
新たな
レコード番号
佐藤
鈴木
イチロー
大地
ピーター
人名テーブルのビュー
レコード番号
①
②
③
④

【図３３】

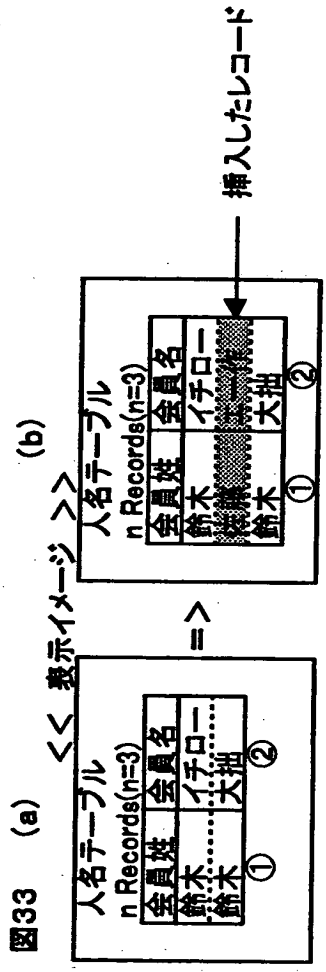


図33
(a)
<< 表示イメージ >>
人名テーブル
n Records(n=3)
全員姓
全員名
鈴木
イチロー
鈴木
大拙
①
②
=>
(b)
挿入したレコード
人名テーブル
n Records(n=3)
全員姓
全員名
鈴木
イチロー
鈴木
大拙
①
②

【図３４】

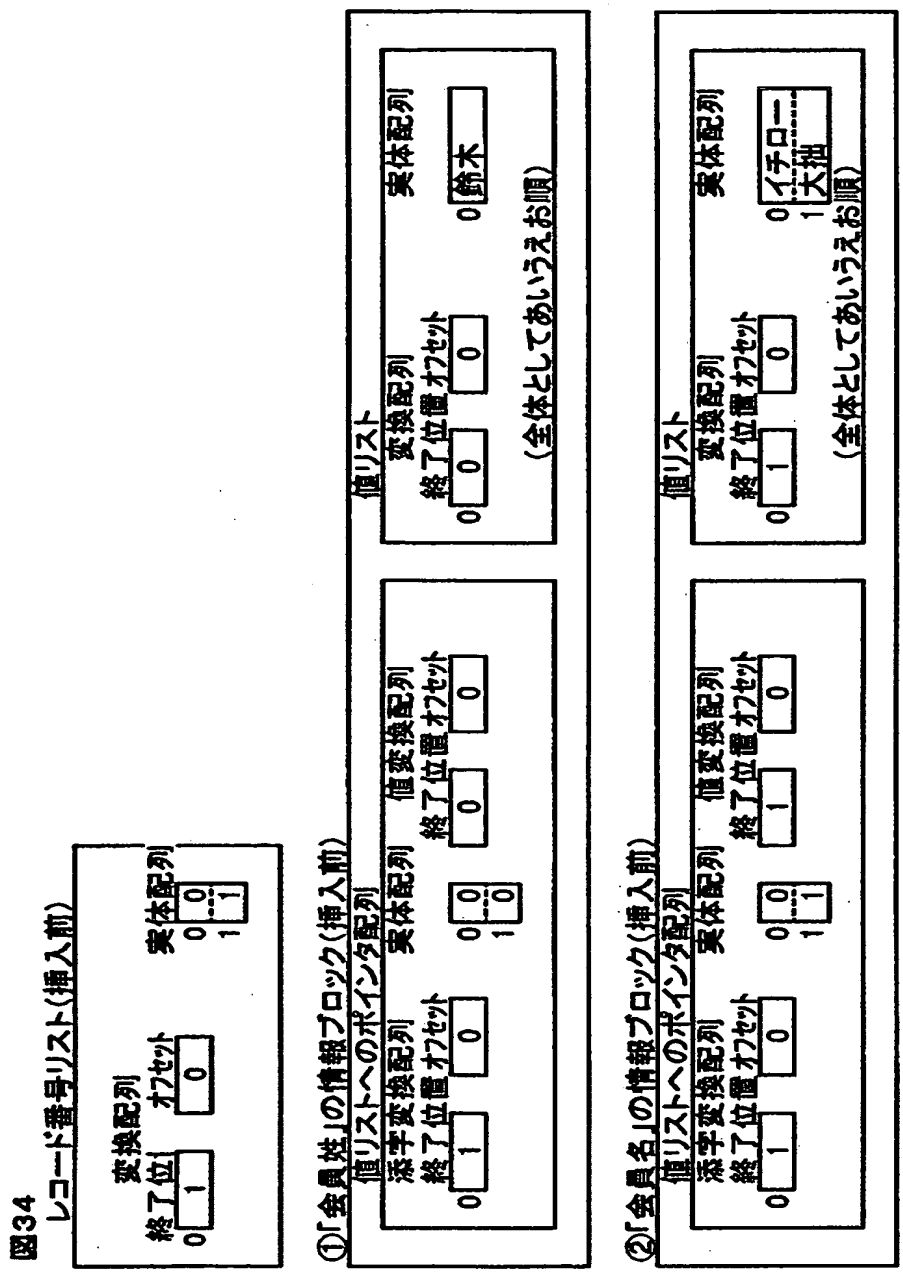


図34
レコード番号リスト(挿入前)
変換配列
終了位置
オフセット
実体配列
①「会員姓」の情報ブロック(挿入前)
値リスト
添字変換配列
値変換配列
(全体としてあいうえお順)
鈴木
②「会員名」の情報ブロック(挿入前)
イチロー
大地

【図３５】

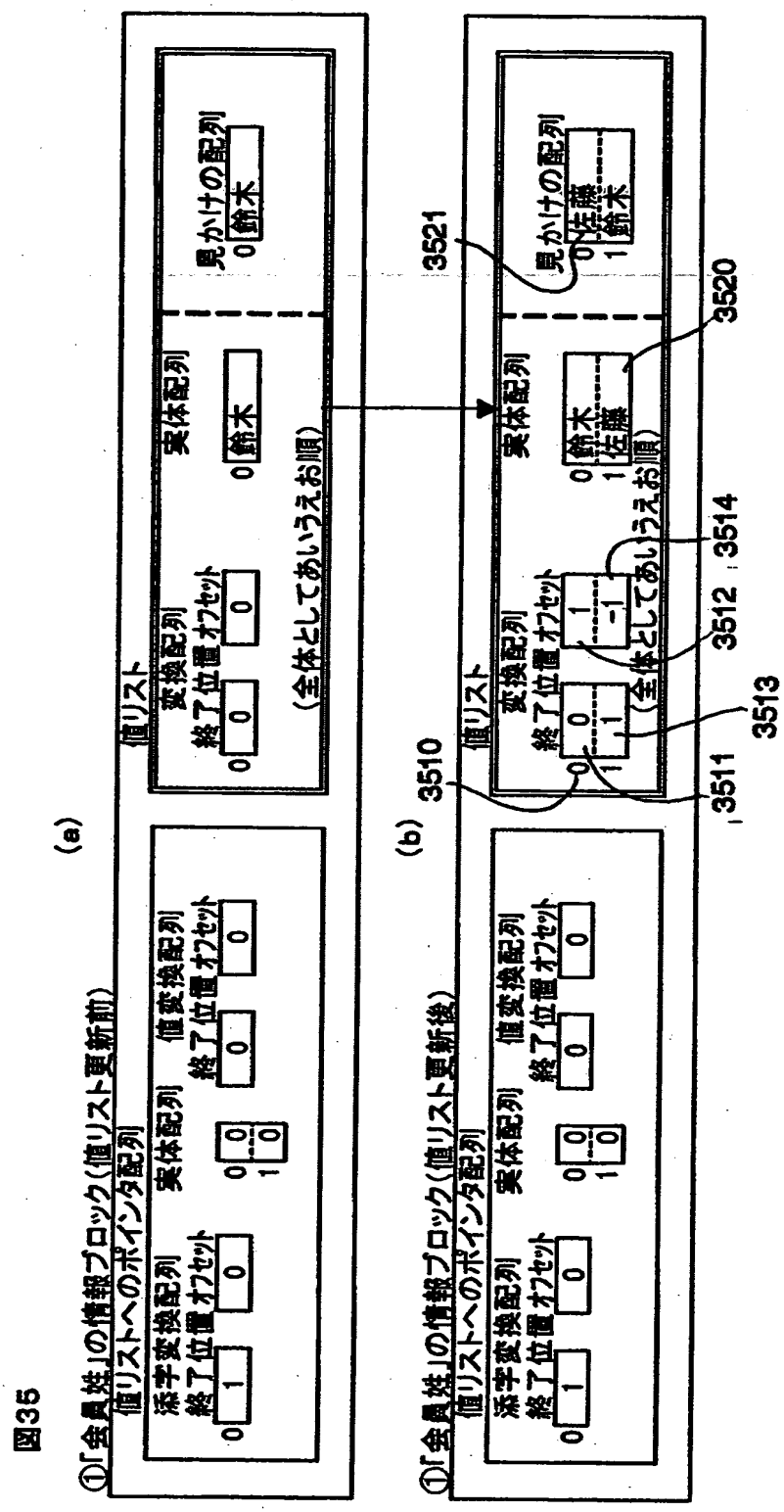


図35
(a)
(b)
①「全員姓」の情報ブロック(値リスト更新前)
①「全員姓」の情報ブロック(値リスト更新後)
値リスト
値リストへのポインタ配列
添字変換配列
終了位置オフセット
実体配列
値変換配列
終了位置オフセット
見かけの配列
(全体としてあいうえお順)
鈴木
佐藤
3510
3511
3512
3513
3514
3520
3521

【図３６】

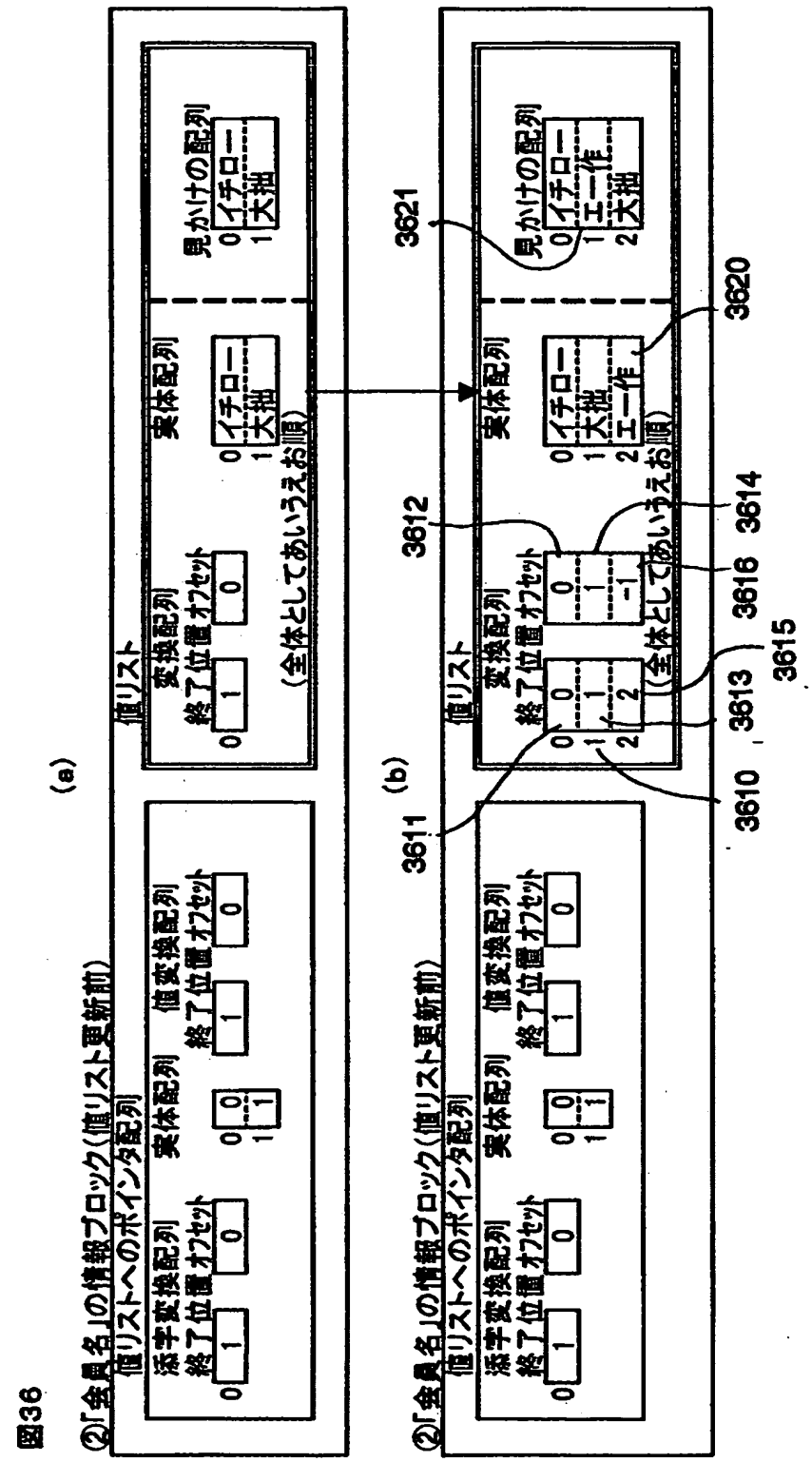


図36
(a)
(b)
②「全員名」の情報ブロック(値リスト更新前)
値リストへのポインタ配列
添字変換配列
終了位置オフセット
実体配列
値変換配列
終了位置オフセット
値リスト
変換配列
終了位置オフセット
実体配列
見かけの配列
0イチロー
1大拙
(全体としてあいうえお順)
0イチロー
1大拙
0
1
0
1
0
1
0
1
0
1
2
0
1
2
0
1
2
0
1
2
0イチロー
1エー作
2大拙
0イチロー
1大拙
2エー作
3610
3611
3612
3613
3614
3615
3616
3620
3621

【図３７】

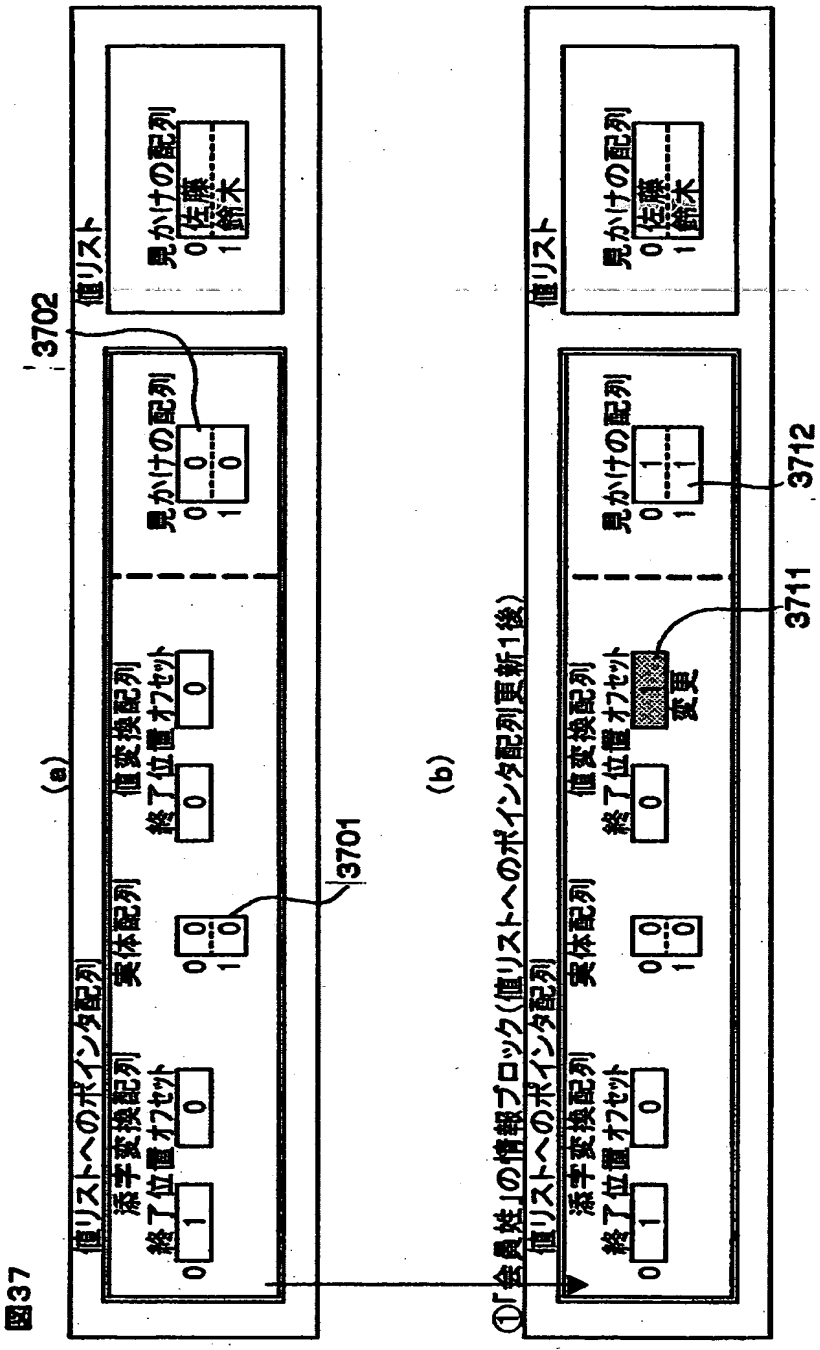


図37
(a)
値リスト
見かけの配列
0 佐藤
1 鈴木
3702
値リストへのポインタ配列
添字変換配列
終了位置オフセット
実体配列
3701
(b)
①「金員姓」の情報ブロック(値リストへのポインタ配列更新1後)
3712
3711
変更

【図３８】

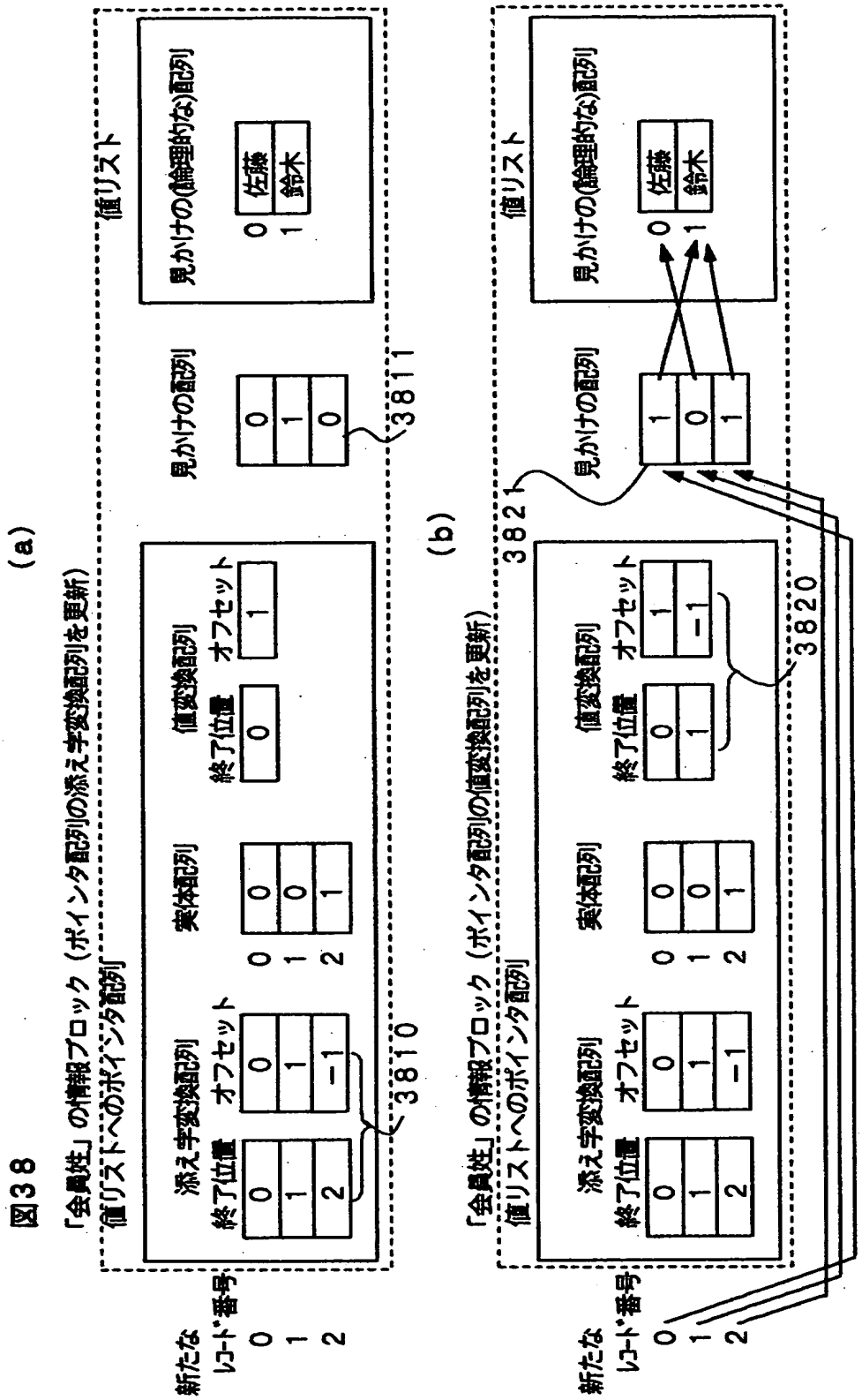


図38
(a)
「全員姓」の情報ブロック（ポインタ配列の添え字変換配列を更新）
値リスト
値リストへのポインタ配列
添え字変換配列
終了位置
オフセット
実体配列
値変換配列
終了位置
オフセット
見かけの配列
見かけの(論理的な)配列
0 佐藤
1 鈴木
3810
3811
新たなレコード番号
0
1
2
(b)
「全員姓」の情報ブロック（ポインタ配列の値変換配列を更新）
値リスト
値リストへのポインタ配列
添え字変換配列
終了位置
オフセット
実体配列
値変換配列
終了位置
オフセット
見かけの配列
見かけの(論理的な)配列
0 佐藤
1 鈴木
3820
3821
新たなレコード番号
0
1
2

【図３９】

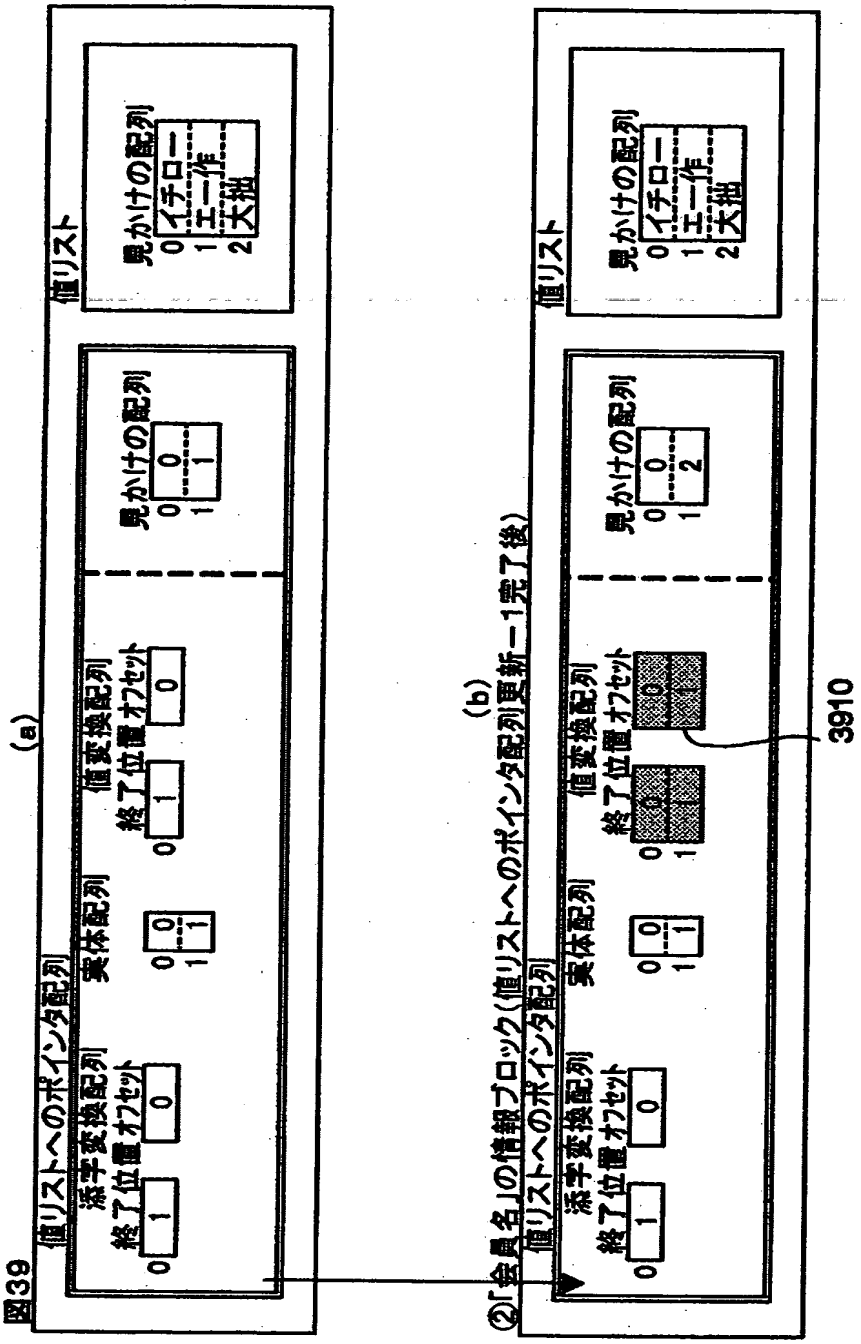


図39
(a)
値リスト
見かけの配列
0 イチロー
1 王一作
2 大拙
値変換配列
終了位置オフセット
0 1
1 0
実体配列
0 0
1 1
添字変換配列
終了位置オフセット
0 1
見かけの配列
0 0
1 1
(b)
②「会員名」の情報ブロック(値リストへのポインタ配列更新－1完了後)
値リストへのポインタ配列
値リスト
見かけの配列
0 イチロー
1 王一作
2 大拙
値変換配列
終了位置オフセット
0 0
1 1
実体配列
0 0
1 1
添字変換配列
終了位置オフセット
0 1
見かけの配列
0 0
1 2
3910

【図４０】

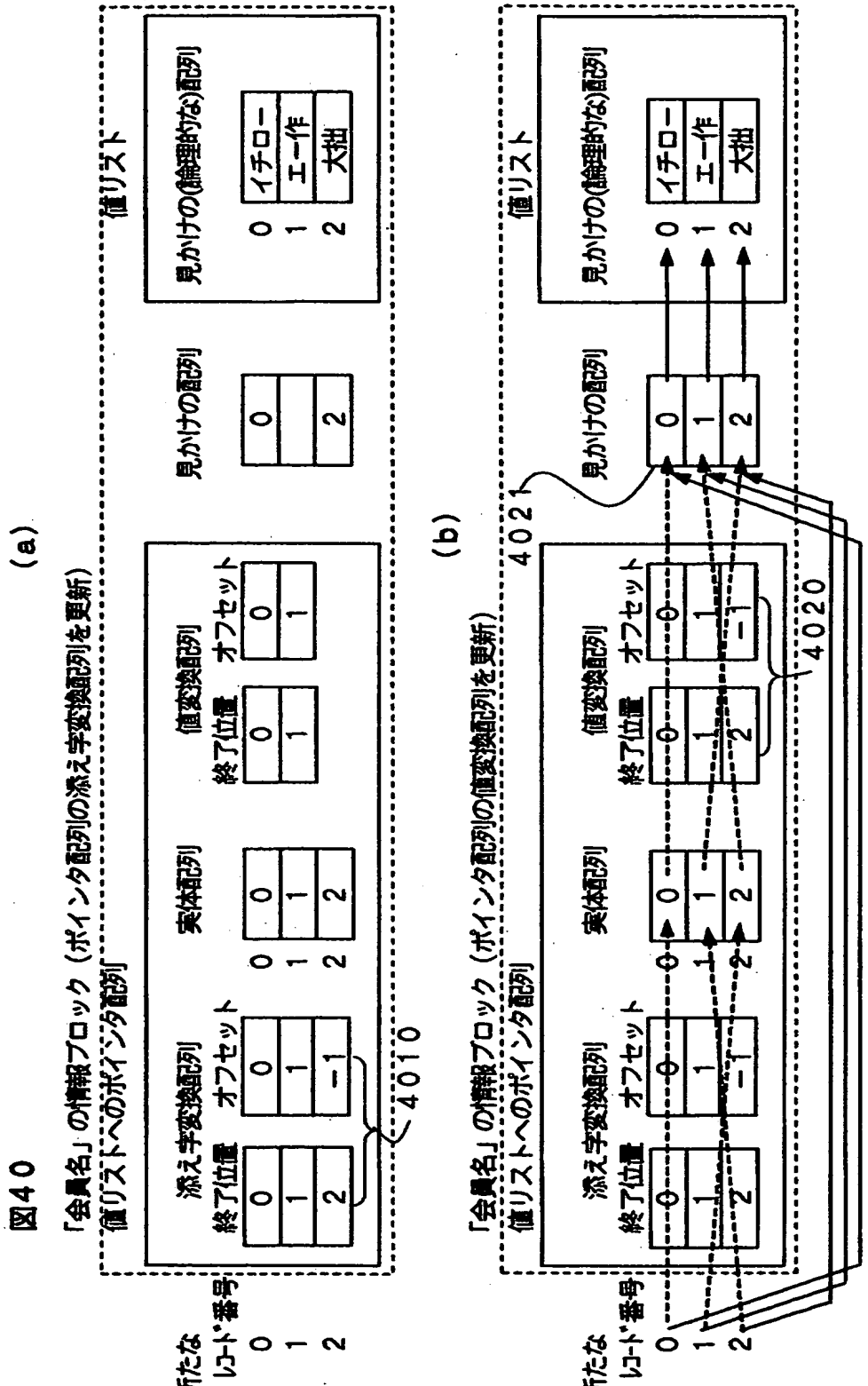


図４０
(a)
「会員名」の情報ブロック（ポインタ配列の添え字変換配列を更新）
値リストへのポインタ配列
値リスト
添え字変換配列
実体配列
値変換配列
終了位置
オフセット
見かけの配列
見かけの(論理的な)配列
イチロー
エーイチ
大拙
4010
新たなレコード番号
(b)
「会員名」の情報ブロック（ポインタ配列の値変換配列を更新）
4021
4020

【図４１】

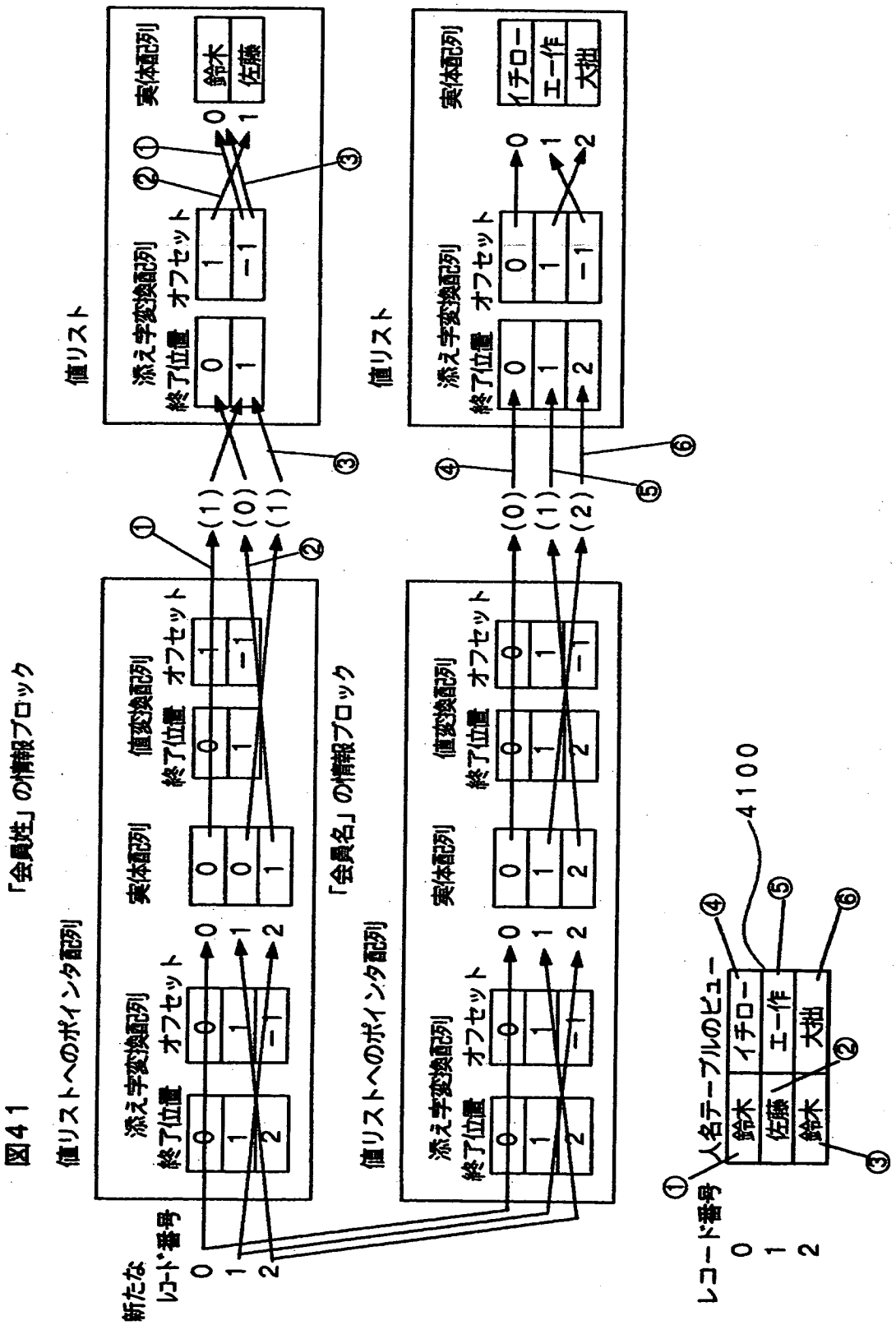


図４１
値リストへのポインタ配列
「会員性」の情報ブロック
「会員名」の情報ブロック
値リスト
添え字変換配列
終了位置
オフセット
実体配列
値変換配列
新たなレコード番号
鈴木
佐藤
イチロー
エーイチ
大地
人名テーブルのビュー
レコード番号
4100

【図４２】

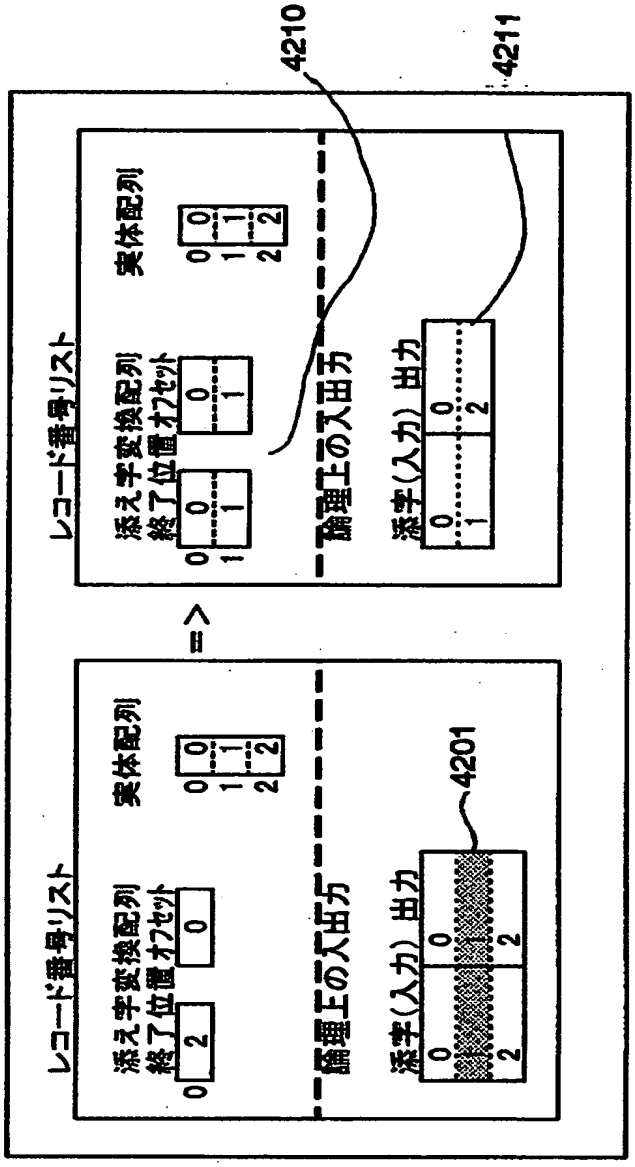
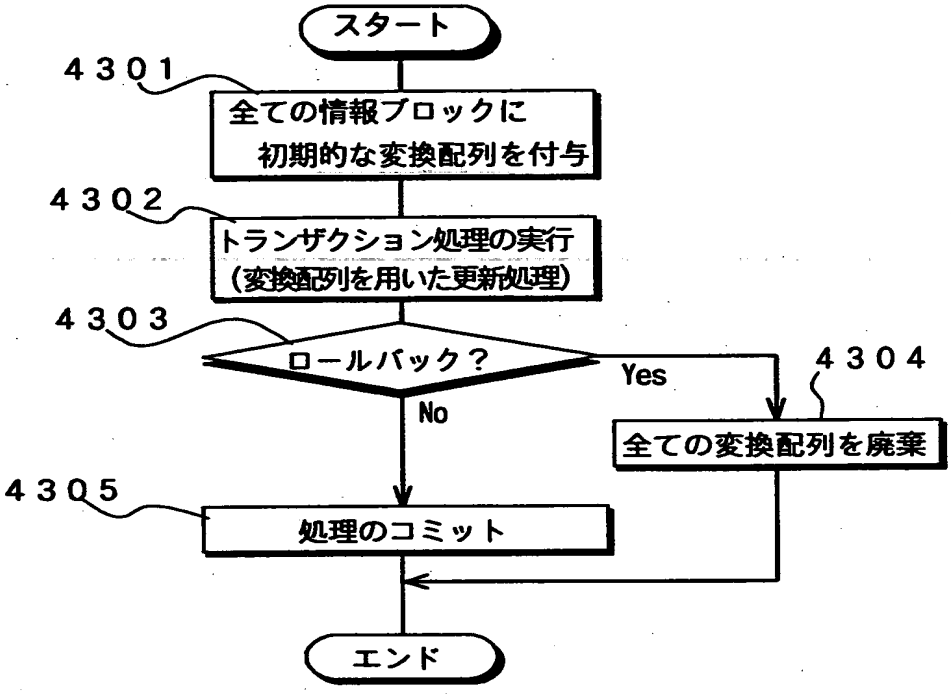

レコード番号リスト
添え字変換配列
終了位置 オフセット
実体配列
論理上の入出力
添字(入力) 出力
=>
4201
4210
4211

図42

【図４３】

図４３


スタート
４３０１
全ての情報ブロックに
初期的な変換配列を付与
４３０２
トランザクション処理の実行
（変換配列を用いた更新処理）
４３０３
ロールバック？
Yes
No
４３０４
全ての変換配列を廃棄
４３０５
処理のコミット
エンド

【図４４】

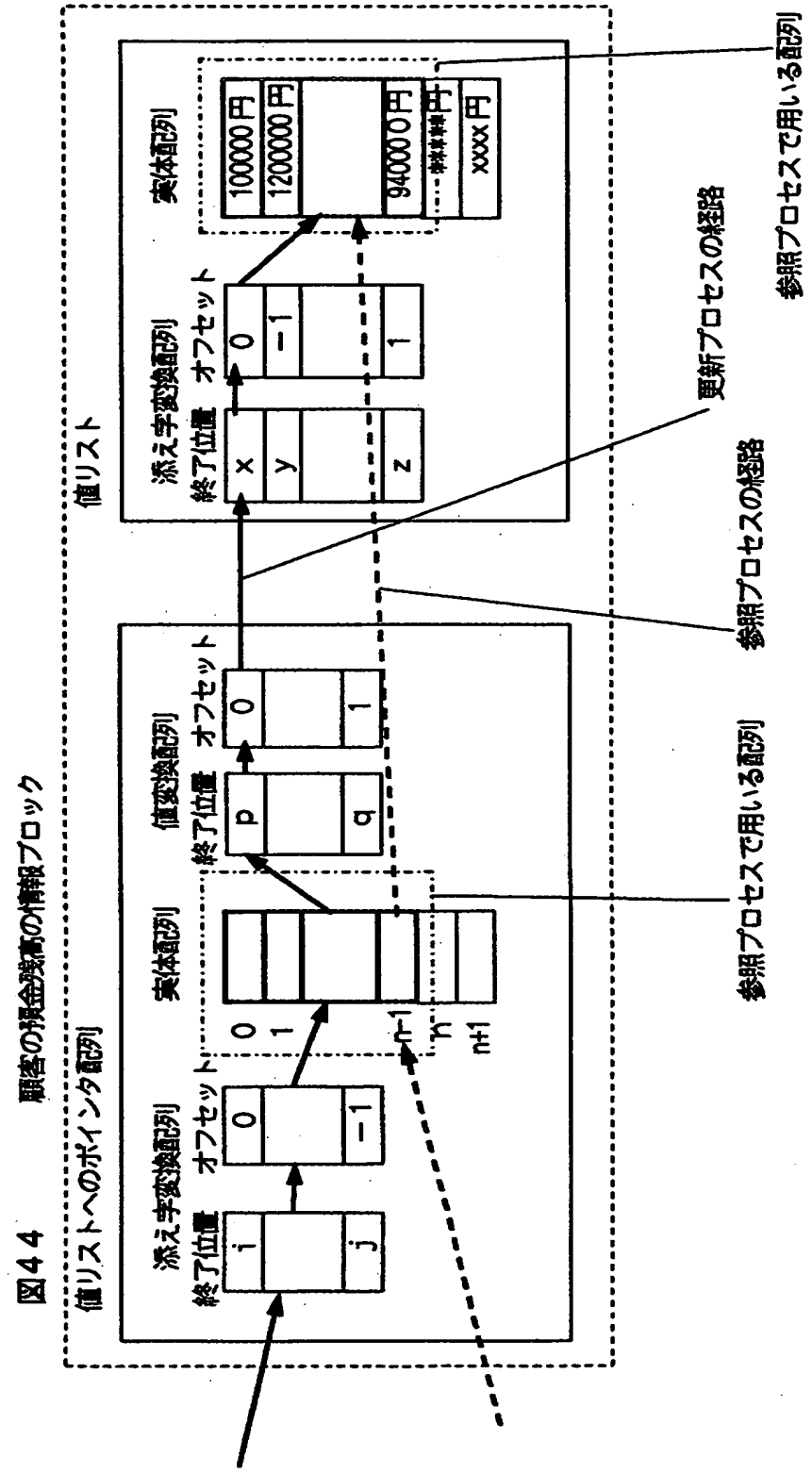
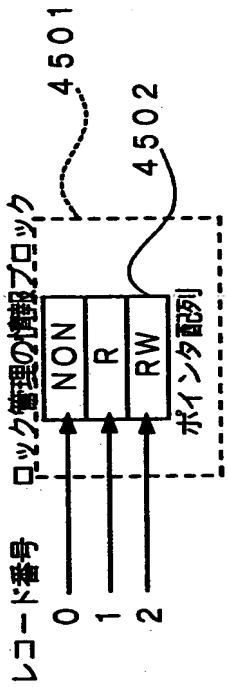


図４４
顧客の預金残高の情報ブロック
値リストへのポインタ配列
値リスト
添え字変換配列
終了位置
オフセット
実体配列
i
j
0
-1
0
1
n-1
n
n+1
p
q
x
y
z
100000円
1200000円
9400000円
xxxx円
参照プロセスで用いる配列
更新プロセスの経路
参照プロセスの経路
参照プロセスで用いる配列

【図４５】



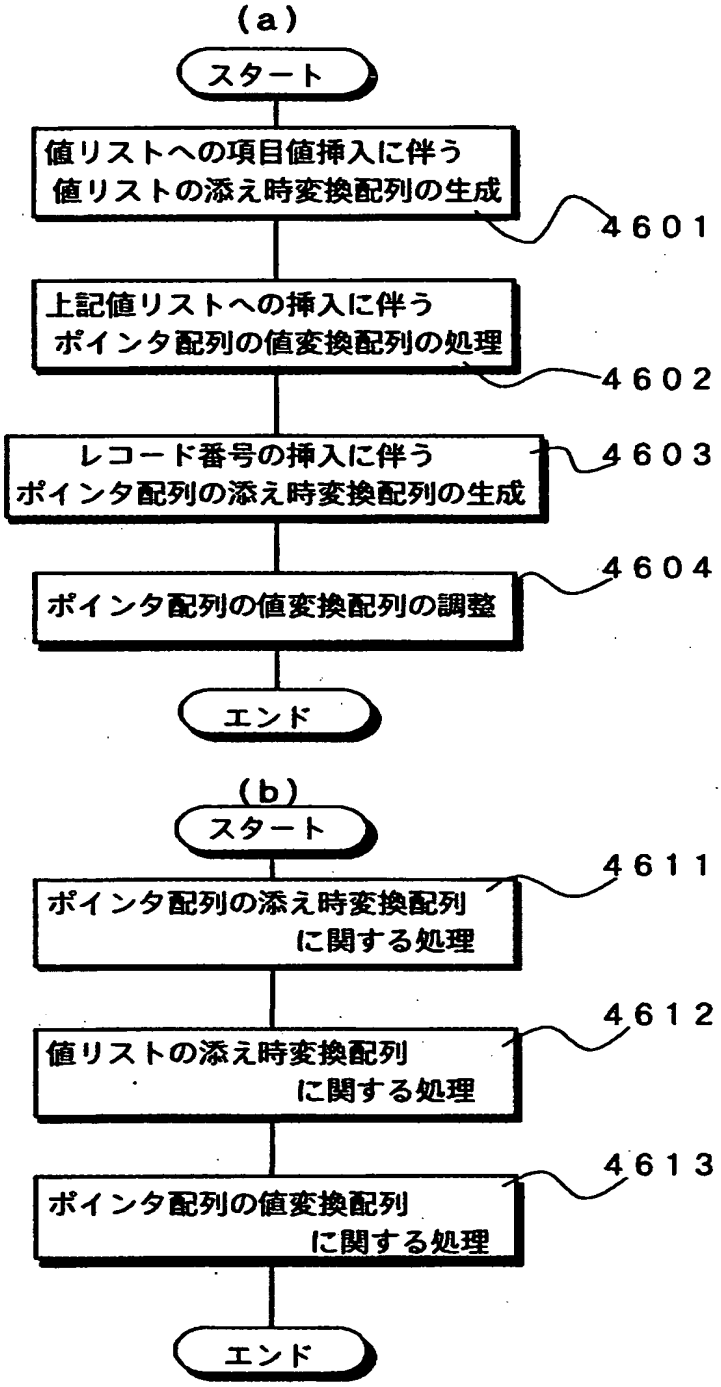
人名テーブルのビュー

鈴木	イチロー
佐藤	エー作
鈴木	大拙

図４５

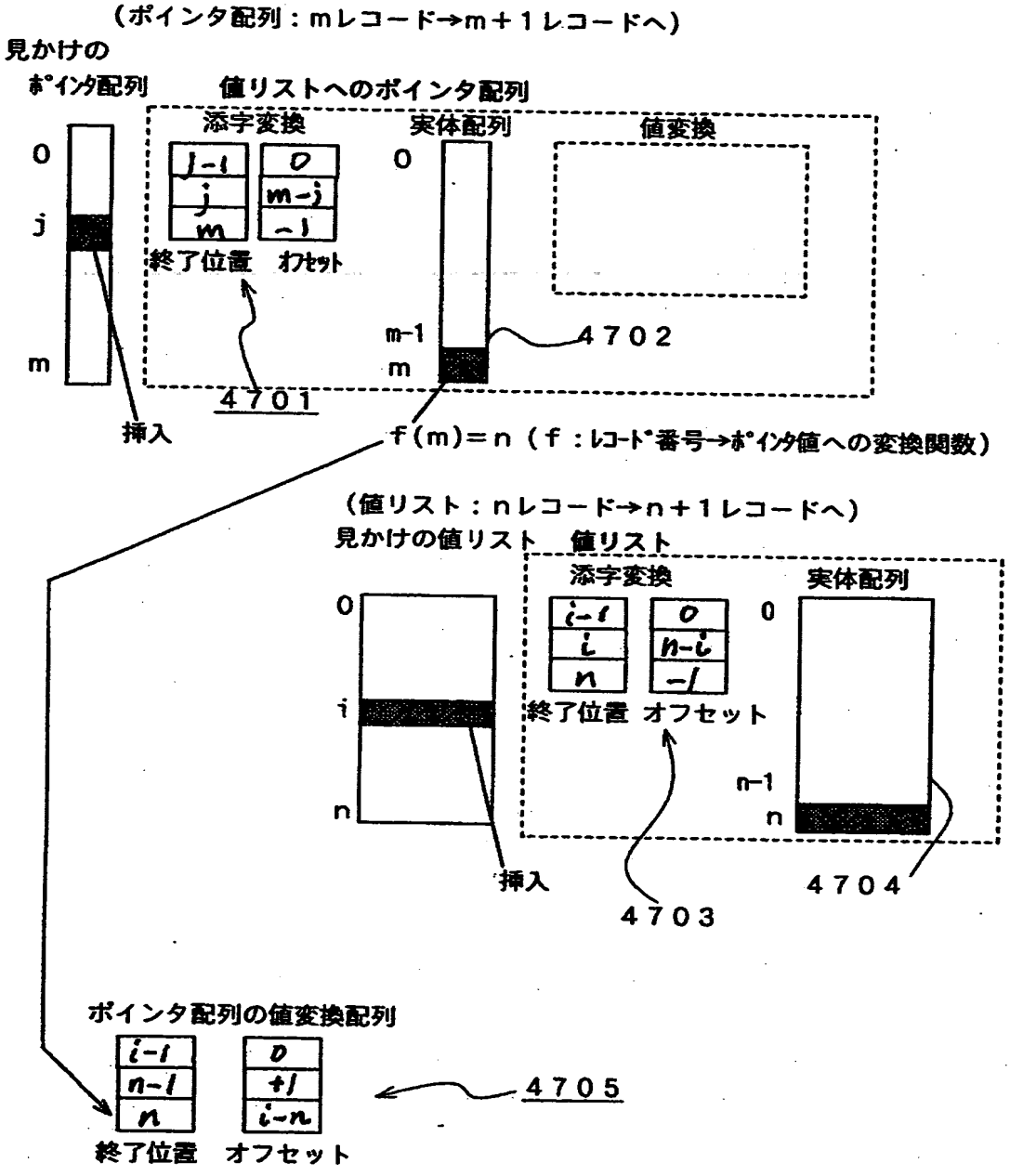
【図４６】

図46


(a)
スタート
値リストへの項目値挿入に伴う
値リストの添え時変換配列の生成
4601
上記値リストへの挿入に伴う
ポインタ配列の値変換配列の処理
4602
レコード番号の挿入に伴う
ポインタ配列の添え時変換配列の生成
4603
ポインタ配列の値変換配列の調整
4604
エンド
(b)
スタート
ポインタ配列の添え時変換配列
に関する処理
4611
値リストの添え時変換配列
に関する処理
4612
ポインタ配列の値変換配列
に関する処理
4613
エンド

【図４７】

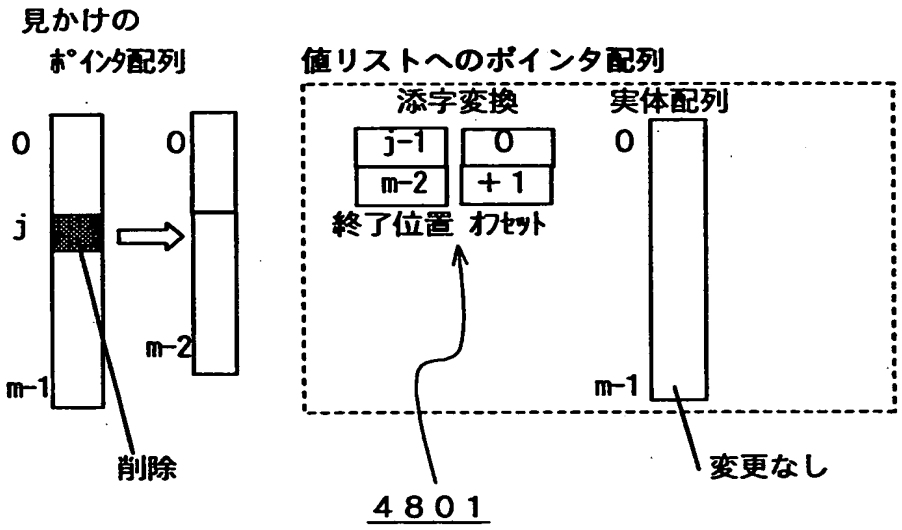
図４７


（ポインタ配列：ｍレコード→ｍ＋１レコードへ）
見かけの
ﾎﾟｲﾝﾀ配列
値リストへのポインタ配列
添字変換
実体配列
値変換
0
j
m
終了位置
ｵﾌｾｯﾄ
m-1
4702
4701
挿入
f(m)＝n（ｆ：ﾚｺｰﾄﾞ番号→ﾎﾟｲﾝﾀ値への変換関数）
（値リスト：ｎレコード→ｎ＋１レコードへ）
見かけの値リスト
値リスト
i
n
オフセット
n-1
4704
4703
ポインタ配列の値変換配列
4705

【図４８】

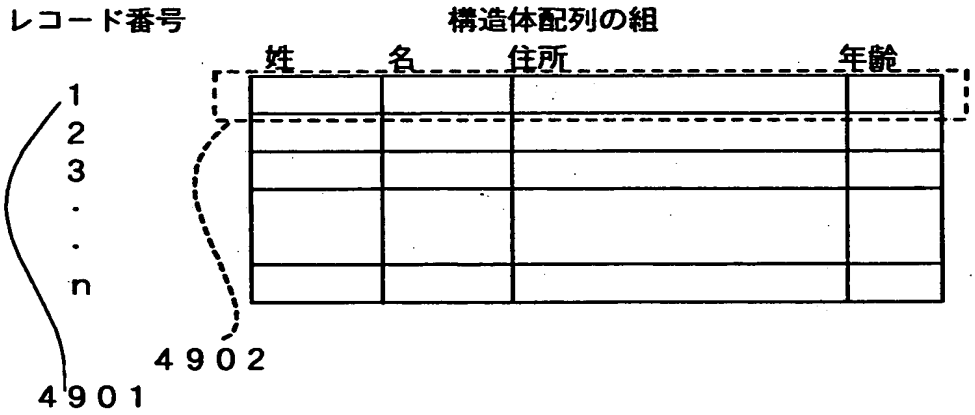
図４８

（ポインタ配列：ｍレコード→ｍ－１レコードへ）



【図４９】

図４９



【図５０】

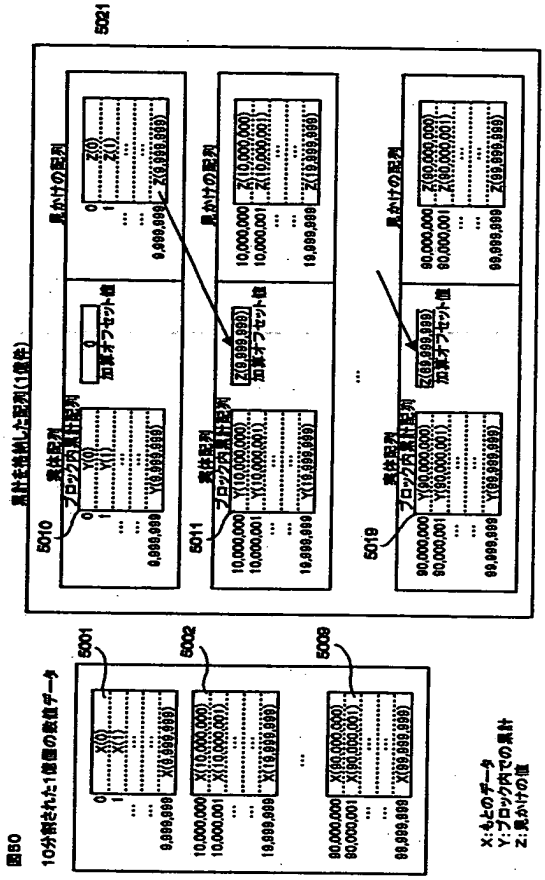

5021
5010
5011
5019
5001
5002
5009
X:もとのデータ
Y:ブロック内での累計
Z:最かけの値

図50

【図５１】

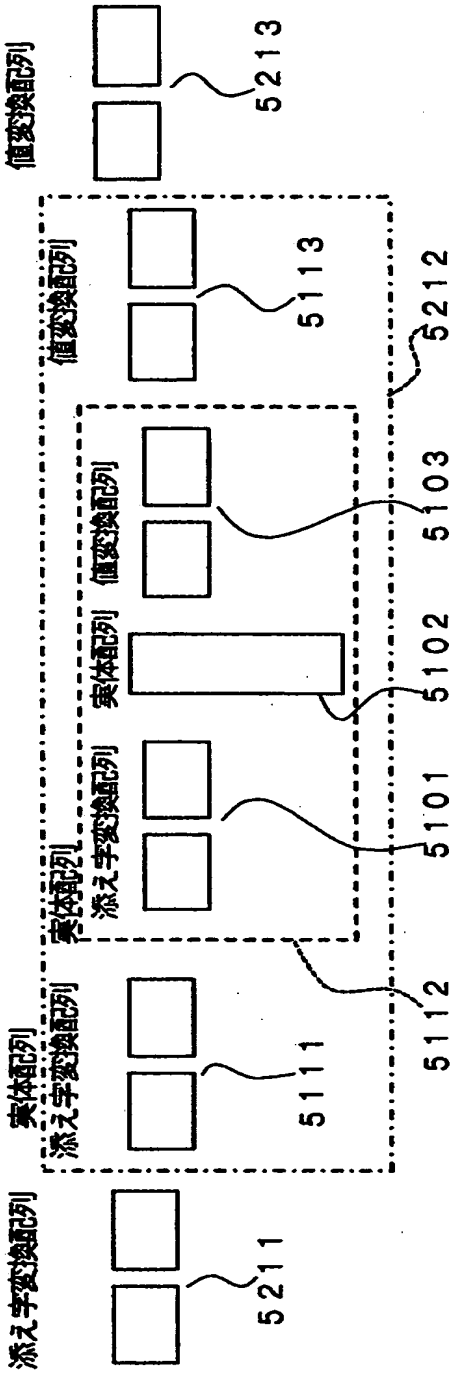
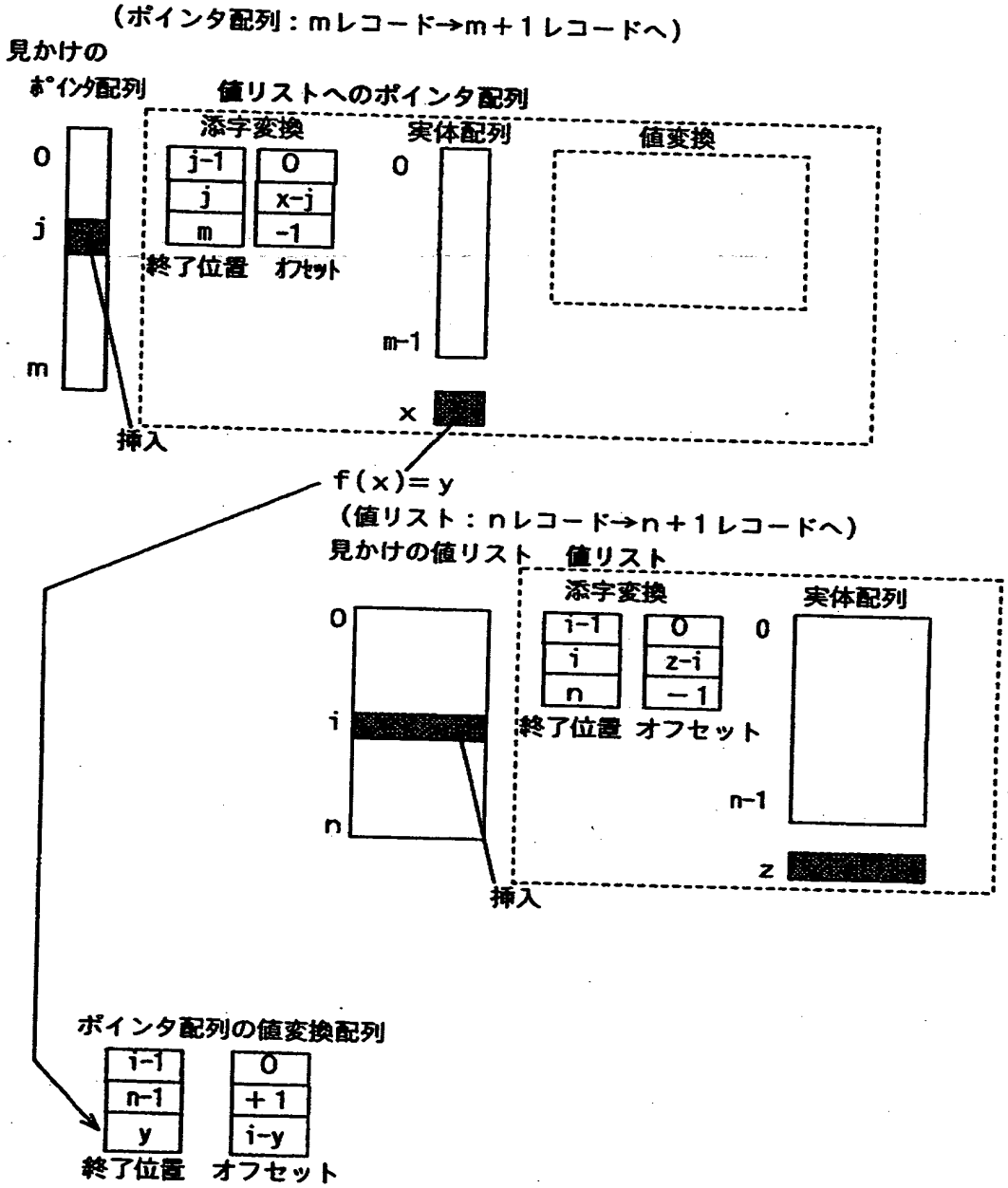

添え字変換配列
実体配列
添え字変換配列
実体配列
添え字変換配列
値変換配列
5211
5111
5101
5102
5103
5112
5113
5212
5213

図５１

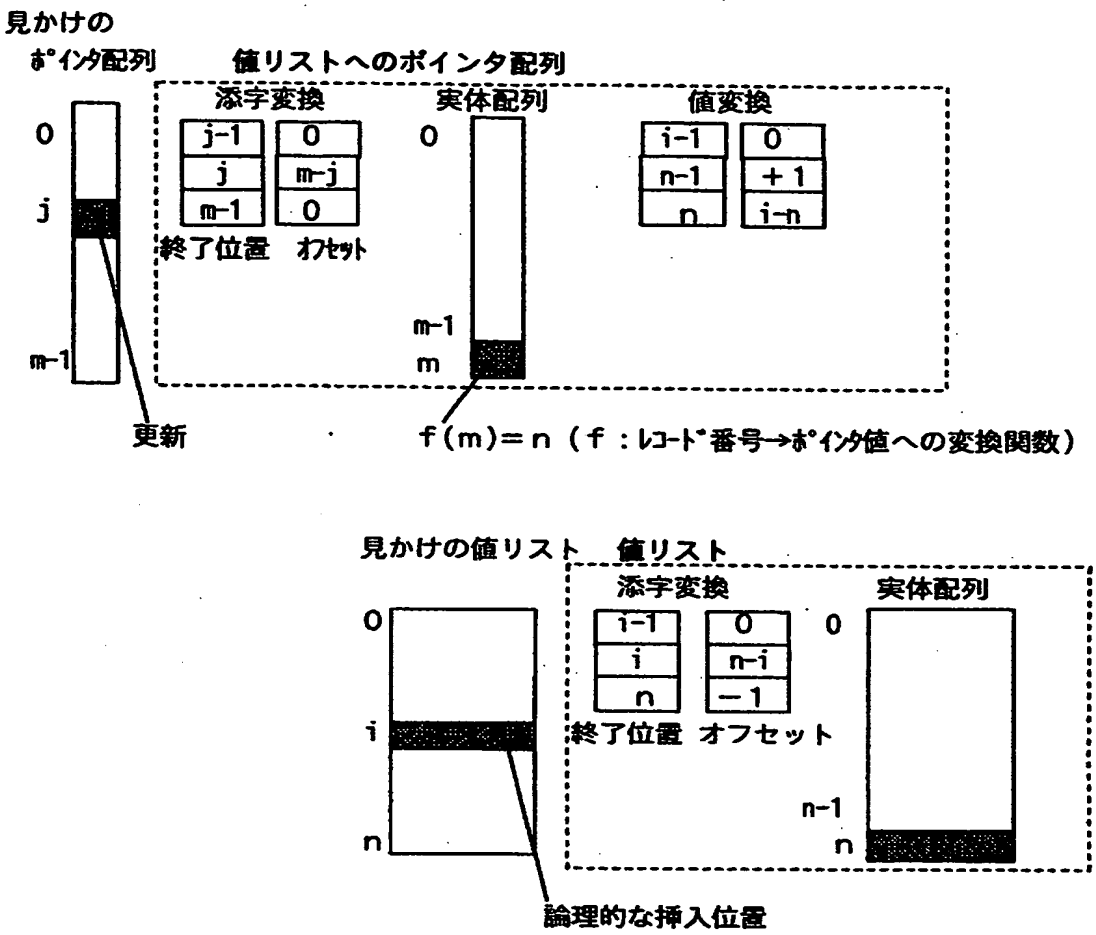
【図５２】

図５２


（ポインタ配列：ｍレコード→ｍ＋１レコードへ）
見かけの
ポインタ配列
値リストへのポインタ配列
添字変換
実体配列
値変換
0
j
m
j-1
j
m
0
x-j
-1
終了位置
オフセット
0
m-1
x
挿入
f(x)＝y
（値リスト：ｎレコード→ｎ＋１レコードへ）
見かけの値リスト
値リスト
添字変換
実体配列
0
i
n
i-1
i
n
0
z-i
－1
終了位置
オフセット
0
n-1
z
挿入
ポインタ配列の値変換配列
i-1
n-1
y
0
＋1
i-y
終了位置
オフセット

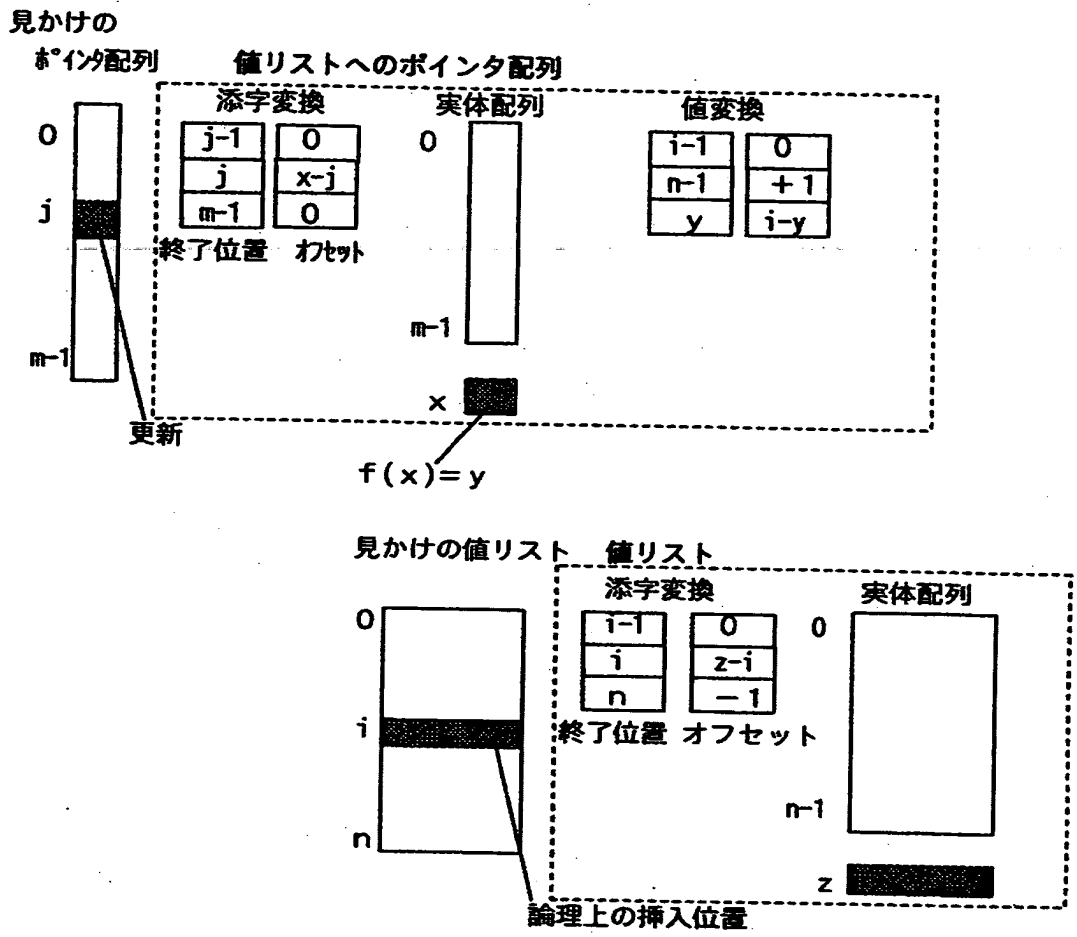
【図５３】

図５３


見かけの
ﾎﾟｲﾝﾀ配列
値リストへのポインタ配列
添字変換
実体配列
値変換
0
j
m-1
j-1
j
m-1
0
m-j
0
終了位置
ｵﾌｾｯﾄ
0
m-1
m
i-1
n-1
n
0
+1
i-n
更新
f(m)＝n（f：ﾚｺｰﾄﾞ番号→ﾎﾟｲﾝﾀ値への変換関数）
見かけの値リスト
値リスト
添字変換
実体配列
0
i
n
i-1
i
n
0
n-i
-1
終了位置
オフセット
0
n-1
n
論理的な挿入位置

【図５４】

図５４


見かけの
ポインタ配列
値リストへのポインタ配列
添字変換
実体配列
値変換
0
j
m-1
j-1
j
m-1
0
x-j
0
終了位置
オフセット
0
m-1
x
i-1
n-1
y
0
+1
i-y
更新
f(x)=y
見かけの値リスト
値リスト
添字変換
実体配列
0
i
n
i-1
i
n
0
z-i
-1
終了位置
オフセット
0
n-1
z
論理上の挿入位置

【書類名】　　要約書

【要約】

【課題】　表形式データにおいて、データの挿入、削除および更新の処理を高速かつ適切になす。

【解決手段】　ＣＰＵ１２は、レコード番号を添え字として受け入れ、当該添え字の範囲に対応するオフセット値を与えるための添え字変換配列を生成し、挿入される項目値の位置を示す挿入位置を特定し、添え字変換配列において、挿入位置に関して、対応する添え字の範囲を確定するとともに、配列の末尾を特定するためのオフセット値を与え、添え字変換配列において、挿入位置に対応するレコード番号より大きな番号を有するものに関して、対応する添え字の範囲をインクリメントするとともに、受け入れられた添え字をデクリメントするためのオフセット値を与え、上記末尾の位置に、挿入すべき項目値を配置し、添え字に、添え字変換配列中の添え字の範囲にしたがったオフセット値を与える。これにより、オフセット値が与えられた添え字を用いて配列中の項目値が特定される。

【選択図】　図１１

認定・付加情報

特許出願の番号	平成１１年　特許願　第２１５４５０号
受付番号	５９９００７３０４７０
書類名	特許願
担当官	第七担当上席　　００９６
作成日	平成１１年　８月１７日

＜認定情報・付加情報＞

【提出日】　平成11年 7月29日

次頁無

出　願　人　履　歴　情　報

識別番号　　〔５９９０７４６４８〕

１．変更年月日　　１９９９年　５月３１日
［変更理由］　新規登録
住　所　　東京都台東区松が谷１－９－１２　ＳＰＫビルディング６０４号
氏　名　　ターボデータラボラトリー有限会社